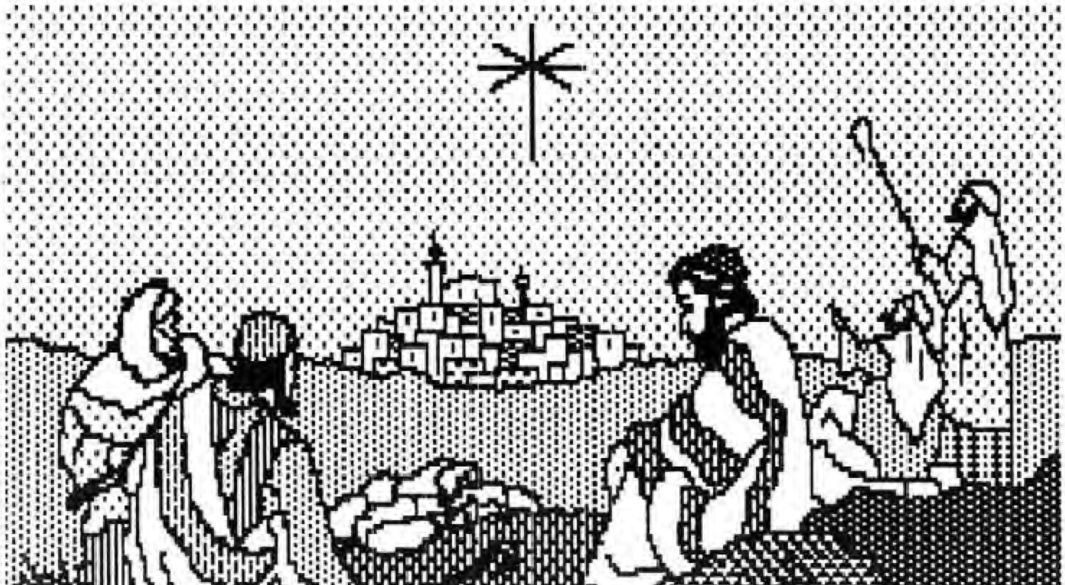
hard core

THE JOURNAL
OF THE
BRITISH APPLE
SYSTEMS
USER GROUP

DECEMBER 1982 £1 VOLUME 2 NO. 6



hardcore

THE JOURNAL OF
THE BRITISH APPLE SYSTEMS USER GROUP
P.O. BOX 174 WATFORD WD2 6NF

Basug Committee

NORAH ARNOLD	Chairman
JOHN SHARP	Secretary
FRAN TEO	Treasurer
JIM PANKS	Membership Secretary
BOB RAIKES	Events Organiser
JOHN ROGERS	Software Librarian

JOHN WELLSMAN, JOHN MARTIN, DAVID BOLTON

Editor	Tony Williams
Assistants	Tony Game, J.P. Williams, Richard Wilday
Advertising	Frances Lowndes

BANKERS

National Westminster Bank Ltd.,
Brompton Square, London SW3 1HH.

The opinions and views expressed are those of the various contributors, and are not necessarily supported by the British Apple Systems User Group.

CONTENTS

INTERNATIONAL APPLE CORE TM

MEMBER OF THE
INTERNATIONAL APPLE CORE

Norah Arnold's Hard Core Cover drawing is a Christmas theme with a difference, commemorating how Apple technology has been used recently to trace the origins of the Star of Bethlehem. In the Mysterious World of Arthur C Clarke, (Yorkshire TV) the doyen of sci-fi writers showed us how one Professor Hughes uses his ITT 2020 to compute the triple conjunction of Pisces before Jupiter and Saturn - I think I've got it right - a much reported phenomenon described from China to Rome as that bright star. It doesn't matter much, but it's nice.

Editorial

Rumour has it that Tuesday, November 2, was the day that certain trusty parties were invited out to Hemel Hempstead to view the Rev E "Super Apple" (64K, fewer chips, upper and lower case) which is set to replace the Apple II when it is withdrawn from production in February, 1993. This news has been so heavily embargoed that everyone knows it. On the other hand it could conceivably happen sooner than that. Since he was not invited along, your intrepid editor instead braved the mist-bound lanes of Bucks that same day to see how Apples were faring out at Castle Priory, Wallingford where the world's most prestigious course on Technology for the Disabled Child was being held.

Apple (UK) Ltd had loaned at last count 13 Apples for the occasion, most of which worked most of the time. This massive presence only confirmed what we already knew, namely that when it comes to the needs of the physically handicapped Apple is tops.

One of the country's foremost workers in this field, Patrick Poon gave a convincing demonstration that his Mackapple package developed at King's College is a powerful weapon enabling children (or adults for that matter) with very poor motor control to communicate via their Apple using simple on-off switch. For the purposes of his demonstration he used an on-off light controller, but it could equally well have been a suck-blow device, a joystick or any simple switch which the handicapped person could operate. With it it is possible to address a keyboard display on the VDU and with great speed and apparent ease print out messages on screen and obtain hard copy. The word processor used is in fact a derivation (with permission) of our old friend the Applewriter. This is not the place to describe the package in detail, but it certainly convinced me that it lends a lot of communicative muscle to the handicapped. At the end of the session Patrick dispensed free copies since King's College has far-sightedly placed it in the public domain. Do we have any members with communications handicaps who would be willing to review this package for Hard Core?

The Castle Priory course was well attended by an amazing cross section of people involved with the disabled, ranging from high-powered computer advisers from Australia through machine code programmers to young and relatively inexperienced teachers and helpers thrown in at the deep end and expected to turn their Apples to immediate good use. The general feeling is that the computer can obviously do an awful lot but when it comes down to it, only they can tell it what to do.

Turning to other matters. Have all our members suddenly been struck with sudden and inexplicable contentment with all aspects of their Apple usage? Have all the bugs flown off to invest the BBC and Spectrum? Judging from the number of letters we are not receiving this must be so. Hard Core thrives on that enjoyable pool of rancour which, for the moment seems to have dried up. So please, members, we know you have problems, tell us about them.

Apology time. An unfortunate omission of a thick black line has resulted in readers assuming that advertiser Francis Teo, accounting consultant, is a part of Lux Computer Services. Not so, I'm afraid. Quite separate geographically and in operations, so apologies to the both of you.

Nottingham SGM

Those of you who could not attend the Special General Meeting in the close vicinity of Beeston College (i.e. the nearby Chequers) on November 13 will be glad to know that your proxy votes contributed to the unanimous decision of the meeting to give the go-ahead to the planned reorganization of BASUG as a company limited by guarantee. The point is, as I am sure you are all bored of hearing us say, that the club should be made liable as a whole for anything that might go wrong rather than the individual members of the committee. However, in practical terms, as far as your membership and benefits are concerned, nothing will change. Unlike the proposal made and rejected at the SGM in Hatfield last year, this does not mean that a separate trading enterprise is to be set up.

Beginners' page

SO YOUR PROGRAM DOESN'T WORK

by John Sharp

Typing programs in from magazines or writing them yourself is not the easiest of jobs at the beginning. So a few tips for a beginner are always welcome.

The most important thing you should do before you start is type NEW. Otherwise you will have a number of lines of another program you don't want. If you have not typed NEW, as you type in your program, you will delete some of the previous program lines by writing lines with the same number. Others you will leave. Your program will then be trying to sort out the logic of the previous program and of yours as well. This is tedious to sort out as you have to go through the listing line by line.

If you are typing from a listing then, providing it is correct, your program will run without any problems. If not there are two possibilities. If you are typing in from a listing and you have confidence that it is a correct listing, then you could go through it character by character to see if you have made an error. If you have done this or you are writing your own program then there are a number of debugging tips you can follow. These are not foolproof, so it is worth looking at common problems with copying listings.

The first thing you should do is run the program. This will tell you that the program does run without syntax errors being present. That is of course providing you access all the lines. Sometimes you might not take a particular option and so miss going through a subroutine or even one particular line. Then it becomes difficult to find that error. If you have done this and there is a problem, you will be presented for example with:-

```
?SYNTAX ERROR IN 45
```

and you could go back to the listing and check line 45 character for character. It could be that a simple mistyping of INPIT instead of INPUT has been made. The most common type of error is a missed character. In particular, the colon ":" and "semicolon ";" are the crucial parts of many lines, so it is worth looking at what they do.
Consider the following program :-

```
10 A = 51
20 PRINT "APPLES ARE"
30 PRINT A
40 PRINT "PENCE EACH"
```

This would print :-

```
APPLES ARE
51
PENCE EACH
```

If you wanted to print
APPLES ARE 51 PENCE EACH
it is necessary to add a semicolon to the end of each line because a semicolon means carry on printing where you left off last.
Normally a program would not present the information on three lines but as a single line :-

```
10 A = 51
20 PRINT "APPLES ARE"; A ; "PENCE EACH"
```

Now consider the case of an input statement :-

```
10 INPUT "HOW MUCH DO APPLES COST ";A
```

This would set the cursor after the question. If the question was put after a PRINT statement as follows, and the INPUT separately then it is still possible to place the cursor waiting for the input next to the question by adding the semicolon:-

```
10 PRINT "HOW MUCH DO APPLES COST ";
15 INPUT A
```

Sometimes it is necessary to do it this way, and continue on the same BASIC line. Thus lines 10 and 15 would come together as

```
10 PRINT "HOW MUCH DO APPLES COST
";:INPUT A
```

Another common error is to leave out the "$" sign at the end of a string variable or put one in when there should not be one. The variable "A" is a numerical variable. The numerical variable "A$" is a string - a set of characters. Mistyping would not give an error of syntax because the computer sees no problem. So the error comes up further in the program. Suppose as follows there is meant to be a string response in line 100 and the following is typed in:-

```
100 INPUT "A RESPONSE ";A

      a lot more program

810 PRINT MID$ (A$,2,1)
```

This time nothing is fed back in the way of errors. When line 810 is reached, an attempt to print the second character of A$ is met with a blank, because although you meant A$ to be created in line 100, it wasn't. A response to a numerical variable was made. If you had tried to enter an alphabetic character then the response

```
? REENTER
```

would be made by the computer.

The opposite problem of adding the "$" when it shouldn't be there gives a response as follows:-

```
200 INPUT "THE NUMBER ";A$
210 B = 20 * A
```

gives a value of zero for variable B, since no value has been assigned to A. If you tried :-

```
220 B = 20 * A$
```

you would be greeted with :-

```
TYPE MISMATCH ERROR IN 220
```

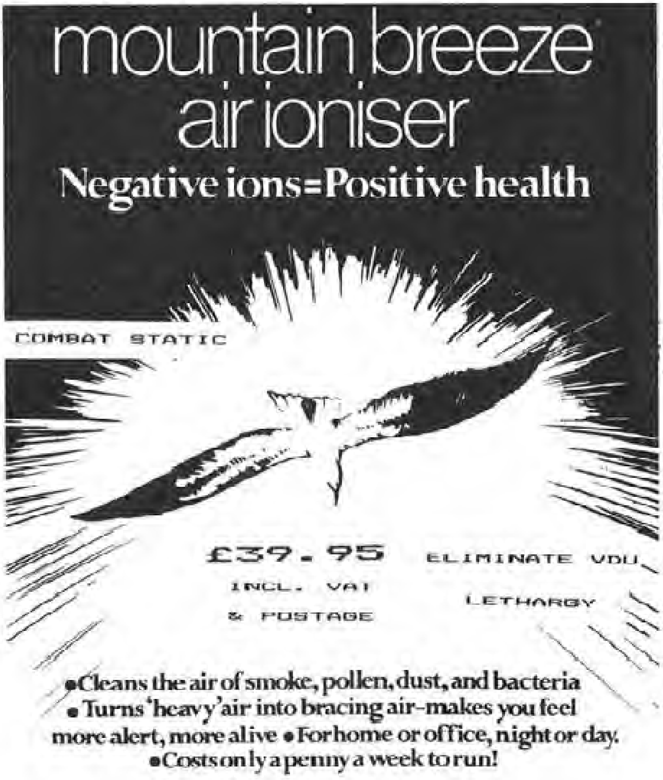
This means the type of variable (a string) is wrong.

This would also arise if the "$" was left off in the following case:-

```
300 ? LEFT$(A,3)
```

Taking the third from the left character of a number is not possible directly. This time there would be a syntax error and so the computer would drop back into immediate mode.

Other ways of debugging will be followed up in the next issue.



THE WILDCARD

MAKES BACK-UP COPIES OF PROTECTED SOFTWARE QUICKLY AND EASILY, WITH JUST THE PUSH OF A BUTTON.

New software locking schemes have rendered even the latest generation of nibble copiers virtually unusable. The bit copiers often require complicated parameter settings, much patience and great effort to use. More often than not the results are disappointing. The **WILDCARD** is different. Rather than copying disks track by track, **WILDCARD** ignores the disk and any copy protection encrypted on it. Instead the **WILDCARD** takes a snapshot of memory in your computer. Now you can make back-up copies of protected software with the push of a button.

FEATURES:

- * Hardware copying device ... push button operation.
- * Copies any 48k (6502) core resident program.
- * No complex parameters are necessary.
- * **WILDCARD** lives in any slot.
- * **WILDCARD** is undectable by software.
- * Produces autobooting disk in 2 minutes.
- * Copies are standard DOS 3.3 disk format.
- * Copies most 64K programs.
- * Copies even the bit copiers!

REQUIREMENTS: 48K + any 16K RAM card + disk drive.

AVAILABLE AT: £99.00 excluding VAT, £113.85 inclusive from

Pete & Pam Computers
New Hall Hey Road
Rossendale
Lancs
BB4 6JG
0706-227011

SBD Software
15 Jocelyn Road
Richmond
Surrey
TW9 2TJ
01-948-0461

Microsource
1 Branch Road
Park Street
St Albans
Herts
0727-72917

In case of difficulty contact :

ELITE SOFTWARE COMPANY,
2 ALMORAH ROAD, HESTON,
MIDDLESEX TW5 9AD
01-572-0453

Chairman's Corner

by Norah Arnold

On Tuesday, October 26th, John Sharp, Jim Panks and myself set off on a prearranged visit to Apple UK. The building was easy enough to recognise, with a huge Apple logo sign outside. Even while parking the car, one couldn't help noticing the number of cars around with two or three Apple stickers on the windows.

After a few minutes wait at reception while we were fitted out with visitors' badges, we were taken through an impressive open-plan office section where several replendent fur coats draped on the backs of chairs confirmed the visible air of affluence.

Cherry Watret, Publicity Director for Apple UK, soon made us feel at ease and we settled down for an exchange of views and opinions. How many members did BASUG have now? How were the local groups doing? As I listened to John Sharp answering these questions I couldn't help thinking that perhaps Apple users were of less interest to Apple UK than non-users. After all, BASUG members already own a machine and unlikely to buy another Apple II in the next few months, whereas the general public outside BASUG can be regarded as potential Apple buyers.

Even so, Cherry soon made it clear that Apple are willing to be of help to BASUG in several ways. If things go well BASUG may be able to use the training room at Apple UK, furnished with about thirty Apple II's, for a meeting or a programming course. This would obviously be a very interesting venue.

Why didn't Apple have a stand at the PCW show? Cherry's answer to this question gave us an insight into the costs which Apple have to bear if they decide to attend a show. If they attend , they cannot get away with doing things 'on a shoestring', so they are bound to calculate which shows will be the most profitable to them and plan to attend those.

On the whole, we had a very amicable meeting and plan to keep in contact in the future. It can only be of benefit to both BASUG members and Apple UK, if a friendly relationship is established and maintained between us.

SOUTHALL WORKSHOP

The one-day workshop in Southall was our first in the London area. The production of a map certainly reduced the grumbles of our first arrivals, compared with the Birmingham event. A number of people arrived at 9.00 a.m., eager not to miss a thing.

First on the agenda was setting up the systems room. There were about a dozen systems up, some in the Visicalc corner, others around the room. Fran Teo soon had the Visicalc clinic going, and this lasted until a late lunchtime.

Meanwhile in the main lecture room, Ian Trackman was giving a talk on 'Good Programming'. This attracted people with a wide degree of experience, from those who had done virtually no programming to those who had their own views on what makes a good program.

The systems room was a hive of activity, with Hal Computers showing a Hard Disk system with a built-in tape back-up, and a plotter which used felt-tip pens to draw graphs and pie charts. Elsewhere much interest was being shown in the 'Arcade Machine' package which a member had brought back from the U S of A. Have you ever tried Galaxian with 2 opposing players at opposite ends of the screen? It certainly looks an interesting package. How about a review, somebody?

By popular request, Ian's talk carried on after lunch for another hour. At the end we had three more names for the Machine Code course in December. There is obviously a need for a course on Basic for Beginners. We'll try to arrange one for the New Year if there is enough interest.

In the afternoon, Peter Turcan gave a talk on data compaction techniques, some of which were used in the Computer Scrabble program which Peter has written. The techniques included those that simply encoded the letters more economically than 8-bit ASCII and also those that depend on the fact that in any language, letters tend to occur only in certain combinations. For example, q is almost always followed by u, so it is therefore possible to replace 'qu' by another code. Peter also gave us a demonstration of his Computer Scrabble program, which many of you may have seen at the PCW show.

In the second systems room representatives of The Last One were demonstrating this much talked about program on the Apple and on the Tandy. There was lively discussion around their machine all afternoon.

Our last scheduled item was a review of the Vista and Mountain Hardware Music systems by John Molloy. The Vista card is much more limited in terms of the range of sounds that can be produced, and in the timbre and pitching of the voices (of which it has 9). The Mountain Hardware music was very impressive for the range of sounds and instruments that it can produce, it being possible to change the instrumentaion after a peice has been entered. Have you ever heard anyone playing Lead Cymbal? It was clear that the impressive sounds heard were not produced without a good deal of hard work and thought.

At 5.00 p.m. we started to pack up, although some were obviously keen to carry on.

It was altogether a very full day with something for Apple users of most interests. Some people went to all the talks, some stayed in the systems room all day. One group arrived with a list of questions about various aspects of hardware and softwere and a pile of work to be done. Others came with no particular plan except to meet with other addicts.

Thanks to all those who came, and especially those who helped by giving talks or demos, took entrance money, or kept an eye on the Literature Library which had been brought along.

FOR SALE

GE Terminet 300 Printer. Letter quality, 30 characters per second serial input. New machine, 110 volt version supplied with transformer. Will print up to 118 characters wide. Complete with very extensive (and expensive!) service manual and cover. £250.00

Epson TX 80B dot matrix printer. 80 characters per second and graphics capability. Friction drive version. Parallel (Centronics) input. Hardly used and in excellent condition. £250.00

Apple II Pal Colour Card £35
Modulator to enable TV to be used as monitor £10
Appleplot disc £5
Applewriter I disks and handbook £10
Books: "Beneath Apple DOS", £5; "What's Where in the Apple", £5; "Applesoft Datafile Programming", £5. ALL AS NEW

Ring P.N. Thurley [redacted]

MERLIN and S-C MACRO

A Comparative Review of Two New Assemblers

by Ian Trackman

What the world of micro-computing needs is software editors. Before you rush, fingers tingling, to your keyboards in order to become the next information technology millionaires, let me explain that software editors are not programs, but a special breed of people.

Probably the biggest mistake, which can be seen wherever you turn in the field of personal software, is to confuse coding with programming. Coding, when done properly, ensures that the computer carries out its instructions correctly. Good programming, on the other hand, acknowledges the existence of human beings and recognises that the computer is their slave (or tool, if you prefer) and not their master. In a word, a good program must be "user-friendly".

A software editor's task is to help a software writer to transform his brilliant ideas into a commercial product. The editor represents the eventual user of the program. He must make sure that the program does what the user is likely to want it to do, that it works properly and that the entire package is user-friendly. Ideally, he should understand the subject-matter of the program, its programming language and the operation of the computer on which it will run. He is independent, diplomatic, persuasive, reasonable, creative, devisive (to find unexpected bugs), nit-picking, logical, objective and able to ensure that all documentation is written in a clear, grammatical and interesting style. My hero is indeed a rare animal.

What does this have to do with a review of two assemblers ? The answer is that one of them shows every sign of not having been edited.

The S-C Macro Assembler is written by Bob Sander-Cederlof and published by S-C Software Corporation. It is sold in the U.K. at £60.00 (plus VAT). The package consists of a disk of software, an instruction manual and a reference card.

Merlin's author is Glen Bredon. For £49.95 (plus VAT), Southwestern Data Systems provide a disk of software and an instruction manual but no reference card (an irritating omission - but more about that anon).

I looked first at the two manuals, starting with S-C Macro. Its cover boldly claims - "Makes assembly language programming on the Apple as easy as programming in BASIC". That opinion is printed in quotation marks, which suggests to me that it is taken from an independent review and yet nowhere was it attributed. Perhaps some-one who doesn't even know the difference between programming and coding prefers to remain anonymous ! It's a ridiculous claim and totally unjustified.

Having been annoyed on page 1, I then turned to the next page to find an Errata Sheet containing 23 corrections to the manual, a mixture of serious omissions and spelling mistakes. A hour or two with my heroic editor would have cured that problem.

The manual begins by explaining how this Assembler is, in fact, the latest version of S-C Corporation's earlier offerings and points out all of its new features. Since I haven't used the previous versions, I wasn't particularly interested. However, if I had written programs with the previous version, this would have been helpful.

Chapter Two is called "Tutorial". In four pages, it contains only the barest rudiments of how to use an editor / assembler. I doubt its value to a novice machine-code programmer. It also mis-described the assembly process.

Subsequent Chapters deal with source programs, commands, directives, operands, macros, 6502 opcodes and Sweet 16. Five appendices cover memory usage, error messages, printer software, customising and a bibliography. One of the appendices summarises the functions of the 6502 opcodes. This would be very helpful to a newcomer to 6502 assembly language.

Merlin's manual contains eight sections relating to Merlin itself, together with a further four sections dealing with the other programs / routines which are included in the package. There is also a section entitled "Product Information", which comprises a series of adverts for the company's other products. Like S-C Macro it has a number of unnecessary typing errors.

To their credit, both assemblers provide the programmer with a comprehensive range of commands, more than sixty in S-C Macro and more than eighty in Merlin. I spent a day working with each assembler in order to write this review and, in that time, I needed to keep referring to their manuals for the correct syntax of the many commands. And that is where both products come in for criticism. Both publishers appear have overlooked the fact that after the first read-through, a programmer will want to turn quickly to the right page to check, say, the parameters of a command or the causes of an error message. Although most of the required information was available in the manuals, neither provided a quick way to find it. S-C Macro at least had a reference card. The text of its manual is badly set out (8 lines to the inch is very difficult to scan). At one point, I forgot how to leave the S-C Macro assembler. I looked in vain in the index under "BASIC", "quit" and "exit". Although better set out, Merlin's manual doesn't even have an index. Time and again, I was forced to thumb through its pages or read the Contents pages until I found the reference that I needed. My editor hero wouldn't have allowed that to happen.

Having grumbled about both sets of documentation, I then took a look at the software. Both assemblers are significantly more advanced than Apple's own DOS Toolkit Editor / Assembler (which I reviewed in Hardcore of March 1981). To allow for more space for source files, Merlin uses an (obligatory) RAM card. S-C Macro makes the choice optional. Using its RAM card version, S-C Macro told me that it had $8600 bytes free for my source code and label tables. Merlin reported $7600 free bytes. Interestingly, the Toolkit offered $7700 bytes with BASIC intact - probably because the editor and the assembler need to be loaded separately into memory and the assembly process must be done via the disk, all of which saves space but is very time-consuming.

Both assemblers provide a macro capability. This means that standard procedures, such as routines to output text from within the user's program, need only be written once and can then be called up, using local parameters, each time that the procedure is needed during assemble. With Merlin, you must remember not to assembly the macro directly (you have to put DO O FIN around it). S-C does this automatically with its ".MA" and ".EM" commands. You can force Merlin to skip macros by altering its configuration parameters and I would have preferred this option to have been the default condition. Both assemblers allow macros to be expanded or repressed in assembly listings.

Both asssemblers include Sweet 16 opcodes and useful notes on how to use them.

Merlin, S-C Macro and the DOS Toolkit all owe their origins to an early prototype called TED+. Consequently, both Merlin and S-C Macro have a number of facilities in common, even though the relevant command syntax is slightly different. DOS and the Monitor are freely accessible. Both systems offer memory management, printer driving, tabbing, searches (with wild cards), formatted listings, replacement, line editing, data definitions, arithmetically based expressions, USR access, conditional assembly, source-file linking, direct assembly to disk for large files, edit-time expression evaluation, 80-column displays and use with lower-case chips. In fact, most of the standard facilities that you would expect from serious, useful assemblers. Source files can be saved and loaded quickly in binary format or loaded and saved as text-files for transfer to and from,other editors / word-processors.

I didn't like S-C Macro's use of line numbering in its editor. Lines of code are actually assigned a unique line-number, rather than having a "floating" number which is re-defined after additions and deletions. Although a fast and versatile re-sequencer and auto-numberer are included in the editor, there is no "move" command, only a "copy" (Merlin has both commands). That results in having two lines of code with the same-line number until the renumberer is used. It also caused an error when I tried to copy a section of code to within itself. Except for the advantage of showing my program's structure by block numbering, I could not see any real benefit to balance the inconvenience of this system.

S-C Macro offers local "re-usable" labels so that you can use, for instance, ".1" as a loop pointer over and over again in programs instead of having to think up new labels for, say, different loops (loop1, loop2, loop3, etc.). Depending on your point of view, this feature either forces you to enter blocks of code at the correct point (good) or creates confusion with duplicate label names (bad). In any event it save space in the symbol table.

Merlin's pseudo-opcodes are three-letter mnemonics, whereas S-C Macro uses two-letter codes preceded by a full-stop. It is useful to have directives which are obvious as such but it is harder to remember - and understand - a two letter code e.g. ".EN" rather than "END".

Provided that you don't re-use the RAM card Merlin will stay there if you return to BASIC. Merlin changes the DOS "INIT" command to "ASSE" and uses the INIT command interpreter area in DOS as a way of switching from BASIC in order to re-connect itself (on the RAM card). You have to add at least one character - "M" is the obvious one in this case - as a pretence file name. It's a useful trick to bear in mind for other applications where you want to put a program on the RAM card and then call it with a DOS-type command from BASIC. I couldn't find an easier way of switching back to S-C Macro from BASIC.

Merlin's manual said that a range of line numbers had to consist of two numbers. In fact "LIST ,100" but not "LIST 100," worked in the same way as in BASIC. In addition, "." lists from the last range given and "/" lists from the last line listed.

S-C Macro allows the expression relators "< = >" whereas Merlin offered "! . &" (meaning EOR, OR and AND). I thought that S-C Macro's BASIC-like expressions would be of more use and certainly more understandable in a listing.

I did some rough-and-ready bench-marking. Using one of my existing programs, I obtained the following very approximate times :-

To find a string at the end of the file :
DOS Toolkit - 15 seconds
Merlin - under 1 second
S-C Macro - under 3 seconds

To find all LDA's :
DOS Toolkit - 17 seconds
Merlin and S-C Macro - 4 seconds

To change all LDA's to XXX :
DOS Toolkit - 25 seconds
Merlin and S-C Macro - 7 seconds

To assemble my program :
DOS Toolkit - over 1.5 minutes
Merlin - 30 seconds
S-C Macro - 27 seconds.

Both assemblers allowed me to enter the "NEW" command (to erase the current file in memory) without asking me to verify the command. Naughty !

Merlin wouldn't accept more than 64 characters in a line. Assembly language programs should be liberally commented and this restriction meant that I had to keep breaking comments up into two or more lines. Furthermore, since it is possible to enter over-length lines in the editing stage, the error only becomes apparent at assembly time. Even then, I would have like a more helpful message than "BAD OPERAND".

I had trouble in both assemblers with lower-case. S-C Macro wasn't flexible enough and worse - a bug - it failed to differentiate between upper- and lower-case in ASCII strings (the bug has been fixed in later versions of the package). Merlin defaulted to lower-case mode at the beginning of every new line - fine for a word-processor but not for an assembler.

S-C Macro is not protected. You can take four back-up copies of Merlin.

Up to this point, I have tried to balance good and bad points between the two assemblers and they come out about even. However, Merlin scores over S-C Macro with its many additional "fringe" commands, such as "INV", "FLS" and "REV" for inverse, flashing and reverse order text. "CHK" creates an assembly-time check-sum and "AST" creates a row of asterisks. I call these "fringe" extras as they add to Merlin's flexibility. On the other hand, they are not glaring omissions from S-C Macro's repertoire. One neat idea that I liked about Merlin was the semi-automatic update option on assembly. It allows you to revise the version number of your source file just before assembly starts.

To differentiate between the two packages, I need to refer back to my champion, the software editor. Apart from my grouse about its documentation, Merlin has been well thought out, is user-friendly and didn't give me unexpected or unwanted results from my commands.

S-C Macro is poor in this respect. A software editor would have made sure that it was properly presented and as bug-free as possible. As it stands, it has numerous defects and cannot match up to Merlin. Even in the short time that I used it, I came across a host of problems. Auto- and manual line-numbering work inconsistently. Lower-case mode was bugged (as mentioned). Any key other than Y or N aborts the "replace" routine. Shift M doesn't work properly in lower-case mode. Auto-line numbering isn't re-set after an assembly. There is no "DDB" or "DW" directive, only ".HS" (hex string) and it won't accept delimiters between pairs of bytes. Unused labels are not flagged in the symbol table. And there were more.

One last point to be made is that both packages contained extra "goodies". There were sample macros and library routines on both disks. In addition, the Merlin package includes a 70-column display routine, a listing formatter and a disassembler and a routine to produce a commented source listing of Applesoft. I understand that a source listing of S-C Macro is also available (price approximately £60.00) and S-C Macro Cross-Assembler on 6800/ 6801/ 6802/ 6809/ Z80/ 68000 (each priced at approx. £30.00).

Conclusions - Both assemblers are, for the serious programmer, a significant improvement over the DOS Toolkit. Merlin is well-written, well-produced and does its task admirably. Subject to the reservations mentioned in this review, and to the need for an obligatory RAM card I strongly recommend it. S-C Macro needs a thorough "wash and brush-up" if it is to present the potential purchaser with a real alternative to Merlin. It is also more expensive. (I understand that it sells in USA for $80, the extra consisting of shipping costs and import duties.)

I am grateful to Mike Glover of Leicester Computer Centre (Merlin) and to Mike Hardwick of Elite Software Co. (S-C Macro) for letting me get my hands on the packages.

BASUG MEDICAL SIG

by Alick Elithorn

The Steering Committee of the BASUG Medical SG which consists of Gordon Jameson (Secretary), Bob Simons (Trasurer) and myself as Chairman together with John Bushman and Gordon Flannigan as Committee Members has met and drafted a Constitution and discussed possible activities. We recognised the difficulty of getting the members of a small group together from distant parts of the country for meetings and one of the functions of the SIG will be to help members organise local groups and local meetings. We thought, therefore, it would be helpful to hold a meeting of the SIG to discuss and plan further activities immediately after, (i.e. at 5.30 p.m.) the Apple Medical Forum which Dr. D.G. Jameson is organising at the Middlesex Hospital on 17th December.
This meeting which is sponsored in part by Apple UK will be a repeat of the very successful one held last January. Essentially the aim is to enbable all those interested in using Apples in medical applications to meet, discuss and learn about what is going on in the medical field. Participants will be encouraged to demonstrate their own software and anyone interested in attending should get in touch with Dr. D.G. Jameson of the Physiology Department at the Middlesex Hospital, Cleveland Street, London W1P 6DP. The forum runs from 10.00 to 5.00 p.m. and there is a registration fee of £18.00 which includes lunch, and overnight bed and breakfast, before or after the meeting, is available for £10.00. The BASUG SIG meeting after the Forum is open to anyone interested whether or not they are participating in the Forum.
In spite of the problems the Steering Committee recognised the need to organise national meetings and Gordon Flannigan has kindly agreed to arrange the first of these at Newcastle at a date yet to be arranged but some time in March or April next year. I would be grateful if anyone willing to organise a meeting in Scotland or the Midlands in the latter part of next year would get in touch with me.

Dark Star Systems

54, Robin Hood Way, Greenford, Middx. UB6 7QN

NOT

just another Apple bit copier

PUT LOCKSMITH BACK ON THE SHELF! SNAPSHOT IS THE BACKUP SYSTEM THAT <u>REMOVES</u> COPY PROTECTION FROM DISKS, AND COPIES MOST PROGRAMS THAT NO BIT COPIER CAN TOUCH – INCLUDING THE BIT COPIERS THEMSELVES!

Have you noticed? Copy protection is getting so sophisticated that many new programs can defeat the bit copiers, or can only be copied after hours of trial and error. But SNAPSHOT will copy any program, in any language, that runs on a 48K Apple II (except for programs that use a Z-80 card or that repeatedly access their own disk while running). **For example, SNAPSHOT will copy every bit copier now sold, in less time than it takes you to read this paragraph.** SNAPSHOT is a peripheral card that uses your Language Card to interrupt a running program and dump the entire contents of 48K and registers to an unprotected backup disk. Unlike bit copiers, SNAPSHOT requires no complex parameter changes or trial-and-error tedium. SNAPSHOT is also ideal for debugging your own programs or analyzing others' programs.

FEATURES:

- Faster and easier to use than any bit copier
- Copies 13-sector programs to 16-sector disks
- Can repeatedly interrupt and resume running any program you want to study or back up
- Full monitor capabilities to examine, modify, trace, single-step, or disassemble an interrupted program

PRICE: £95.00 (inclusive) - discounts on quantity orders.
Please make cheques payable to Dark Star Systems.

Tel.: (01) 900 0104

Prop.: S.L.Johnson

Printing Low Resolution Graphics

by Philip Boit

A number of printers have a bit addressable graphics mode which will print a dot pattern corresponding to the 1 bits of the ASCII code of the transmitted character., i.e. the non-zero bits selectively trigger the wires of the print head.

Therefore it is possible to define "colour" strings which, when printed in this way, will simulate the monochrome display "colours of an LRG display. The simulation will not be exact as the VDU display contains "toggled" dots which cannot be matched by a printer, but this is also the case with some HGR displays.

The procedure is to define an array of "colour" strings using actual characters or the CHR$ function, scan the LRG display using the SCRN function and then print out the dot pattern corresponding to the colour of each block of the display. This approach would allow easy reversal of the display, a left to right translation or a rotation through 90 degrees if required.

In practice there are some problems.

If nothing is done to stop it, the Apple will force a dot pattern corresponding to CR, LF and "space" into the print at odd intervals and, in each case I've tried, Paper Tiger 440, Epson MX-100 and MPI 99G the instructions given in the printer manual had to be modified to stop this. In fact the Paper Tiger routine still has two defects. The bit pattern of an LF is printed at the start of each line and a CR, LF "space" will be forced if there is a continuous horizontal strip of printing with no gaps in it.

The second problem is that, the more versatile your printer, the more control characters it is able to recognise and the likelihood increases that a byte intended to print a dot pattern will activate another printer function. Not all of the printer's control characters will be active in the bit addressable graphics mode but it was a problem, particularly with the Epson routines.

The listing shown is for a facsimile print for the Paper Tiger using a 6 x 7 dot array for each LR block. If the routine is to be used more than once the string array should be dimensioned and defined in the program initialisation section.

In fact I find it more suitable to use this process to print various hatchings rather than attempt to match the VDU display and use 6 x 6 arrays of dots as these butt together to give continuous patterns.

The operation of the printer wires, etc. varies from machine to machine and I've included brief notes for the three machines with which I have used this process.

1) Paper Tiger 440

The Paper is not one of the best machines for this process as the vertical alignment, at least on our machine, is not good.

There is a fixed vertical tab of 6 dots so only six wires can be used. The bit code is interpreted as bit 0, top wire to bit 5, 6th wire, i.e., codes in range 0 to 63 can be used.

2) Epson MX-100

The vertical tab can be controlled. The manual says that eight of the nine wires can be activated but I have not been able to persuade the Apple to send a character with the 8th. bit set. I use the first 6 bits which allows the same range of values as for the Paper Tiger but these are interpreted in the reverse order i.e. bit 0 activates the 8th wire, bit 5 the 3rd wire.

The Epson can also be set into the double density mode which gives very sharp prints and could be used to simulate toggled dots of the VDU display.

3) MPI 99G

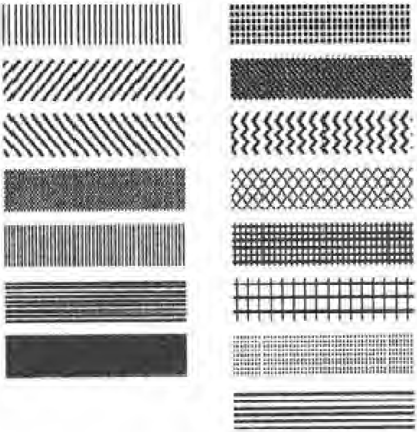
The vertical tab can be controlled.

Only six of the seven wires can be activated but bit 6 has always to be set to 1, thus giving a range of codes from 64 to 127.

The bits are interpreted as bit 0, bottom wire to bit 5, second wire.

```
1000 S$ =  CHR$ (0): DIM C$(15): REM
                       'COLOUR' STRINGS
       ARRAY
1010 C$(0) = S$ + S$ + S$ + S$ +
     S$ + S$ + S$:C$(1) = "*" + S
     $ + "*" + S$ + "*" + S$ + "*
     "
1020 C$(2) = C$(1):C$(3) = "?" +
     S$ + "?" + S$ + "?" + S$ + "
     ?":C$(4) = "*" + S$ + "*" +
     S$ + "*" + S$ + S$
1030 C$(5) = "*******":C$(6) = "?
     " + S$ + "?" + S$ + "?" + S$
      + "*"
1040 C$(7) = C$(3):C$(8) = C$(4):
     C$(9) = "*" + S$ + "?" + S$ +
     "?" + S$ + "?"
```

```
1050 C$(10) = C$(5):C$(11) = C$(3
     ):C$(12) = "?" + S$ + "?" +
     S$ + "?" + S$ + S$
1060 C$(13) = C$(9):C$(14) = C$(6
     ):C$(15) = "???????"
1070  PR# 1: PRINT  CHR$ (3)
1080  FOR J = 0 TO 39
1090  FOR K = 0 TO 39
1100  PRINT C$( SCRN( K,J));
1110  NEXT K: PRINT  CHR$ (3); CHR$
     (11);: PRINT  CHR$ (3); CHR$
     (13);
1120  NEXT J: PRINT  CHR$ (3); CHR$
     (2): PR# 0
1130  RETURN
```



```
1000  DIM C$(15): REM        DD I
     S DOUBLE DENSITY FLAG, ON=1
1010 A$ =  CHR$ (0):B$ =  CHR$ (6
     3):C$ =  CHR$ (42):D$ =  CHR$
     (21):E$ =  CHR$ (36)
1020 F$ =  CHR$ (9):G$ =  CHR$ (1
     8):H$ =  CHR$ (54):I$ =  CHR$
     (B)
1030 J$ =  CHR$ (17):K$ =  CHR$ (
     34):L$ =  CHR$ (20):M$ =  CHR$
     (1):N$ =  CHR$ (27)
1040 O$ =  CHR$ (24):P$ =  CHR$ (
     48):Q$ =  CHR$ (33):R$ =  CHR$
     (3):S$ =  CHR$ (6):T$ =  CHR$
     (12):U$ =  CHR$ (45)
1050 C$(0) = A$ + A$ + A$ + A$ +
     A$ + A$
1060 C$(1) = B$ + B$ + B$ + B$ +
     B$ + B$
1070 C$(2) = B$ + A$ + B$ + A$ +
     B$ + A$
1080 C$(3) = C$ + C$ + C$ + C$ +
     C$ + C$
1090 C$(4) = C$ + D$ + C$ + D$ +
     C$ + D$
1100 C$(5) = O$ + P$ + Q$ + R$ +
     S$ + T$
1110 C$(6) = T$ + S$ + R$ + Q$ +
     P$ + O$
1120 C$(7) = E$ + E$ + E$ + E$ +
     E$ + E$
1130 C$(B) = B$ + A$ + A$ + B$ +
     A$ + A$
1140 C$(9) = C$ + A$ + C$ + A$ +
     C$ + A$
1150 C$(10) = I$ + I$ + I$ + B$ +
     I$ + I$
1160 C$(11) = G$ + B$ + G$ + G$ +
     B$ + G$
1170 C$(12) = K$ + L$ + I$ + L$ +
     K$ + M$
1180 C$(13) = O$ + P$ + Q$ + P$ +
     O$ + T$
1190 C$(14) = N$ + H$ + U$ + N$ +
     H$ + U$
1200 C$(15) = H$ + H$ + A$ + H$ +
     H$ + A$
1210  IF DD THEN  GOSUB 1340
1220  PR# 1
1230  PRINT  CHR$ (27); CHR$ (65)
     ; CHR$ (6);
1240  PRINT  CHR$ (27);"U"
1250  FOR J = 0 TO 39
1260  FOR M = 0 TO 1 + 2 * DD
1270  PRINT  CHR$ (27); CHR$ (75 +
     DD); CHR$ (120); CHR$ (0);
1280  FOR K = (20 - 10 * DD) * M TO
     (20 - 10 * DD) * M + 9 + 10 *
     (1 - DD)
1290  PRINT C$( SCRN( K,J));
1300  NEXT
1310  NEXT : PRINT  CHR$ (13);
1320  NEXT : PR# 0
1330  RETURN
1340  FOR J = 0 TO 15:L$ = ""
1350  FOR K = 1 TO 7:L$ = L$ +  MID$
     (C$(J),K,1) +  MID$ (C$(J),K
     ,1): NEXT
1360 C$(J) = L$
1370  NEXT
1380  RETURN
```

Tabby Cat

by Philip Bolt

If a printed CATALOG listing is used as part of a disk management system it is inconvenient to have to press a key every 22 items and irritating that there is no left margin to allow easy filing of the hard copy.

Both of these minor problems can be cured by use of the routine listed below.

The necessity for a key press is removed by POKing an RTS,(96), into $AE34,(44596), Ref, "Bag of Tricks" p6-9, "Beneath Apple DOS" p7-3.

Tabulation of the CATALOG listing is carried out by using an alternative to COUT as output routine in two places, $ADB4/5 and $ADE6/7,(44468/9 & 44518/9), which sets the horizontal cursor position before using COUT. This routine is patched into an unused area of DOS at $BA69,(47721). The 'END routine' resets DOS to normal operation.

The memory dump from $BA69 shows how the routine works.

The routine could be made into a permanent DOS patch (Ref. Beneath Apple DOS, Ch. 7) thus requiring only the POKE's to 44468/9 and 44518/9 to activate it but I prefer to leave the unused area for other temporary patches.

```
2999  REM
***** MODIFY DOS FOR PR
     INTING *****

3000  POKE 44596,96: POKE 44468,1
     05: POKE 44469,186: POKE 445
     18,105: POKE 44519,186
3010  FOR J = 47721 TO 47740: READ
     I: POKE J,I: NEXT
3020  DATA 141,128,186,165,36,201
     ,0,208,4,169,5,133,36,173,12
     8,186,32,237,253,96

2550  REM

***** END ROUTINE *****

2560  POKE 44596,206: POKE 44468,
     237: POKE 44469,253: POKE 44
     518,237: POKE 44519,253

BA69-    8D 80 BA     STA    $BA80
BA6C-    A5 24        LDA    $24
BA6E-    C9 00        CMP    £$00
BA70-    D0 04        BNE    $BA76
BA72-    A9 05        LDA    £$05
BA74-    85 24        STA    $24
BA76-    AD 80 BA     LDA    $BA80
BA79-    20 ED FD     JSR    $FDED
BA7C-    60           RTS
BA7D-    A0 82        LDY    £$82
BA7F-    B3           ???
BA80-    B3           ???
BA81-    C5 AA        CMP    $AA
BA83-    A0 82        LDY    £$82
BA85-    C5 B3        CMP    $B3
BA87-    B3           ???
BA88-    AA           TAX
BA89-    88           DEY
BA8A-    82           ???
BA8B-    C5 B3        CMP    $B3
```

pascal pages

DISK DIRECTORIES FROM WITHIN UCSD PASCAL PROGRAMS

by C Brandon Gresham Jr.

Quite often when I am running a program I will want to see what files are on the disks currently on-line. This is relatively easy under Apple DOS but it wasn't intuitively obvious to me how to do it under the Pascal Language system. I knew that the directory was on the disks in an orderly fashion. I also knew that, given a little effort, once I knew that order, I could arrange to see all the information my heart desired - about the files on the disk, that is. However, since I have a thousand and one projects going I kept waiting for someone to tell me what the missing magic command was.

I still don't know what that magic command is, but I ran across a program fragment in the December issue of the Apple-Dayton Newsletter (Dayton, Ohio) which had all the information I needed on the order of information in the directory.

As a result I sat down and created the program which is listed here. In reading the listing you should notice:

1) The constants are set for an Apple running UCSD Pascal and using 5" drives. Notice that the last constant DIR_BLK tells you where to look for the directory.

2) The type declaration for DIR_ENTRY describes what is found for each entry in the disk directory. There are two types of entries that satisfy nine different file types, viz. the CASE statement.

3) In both the SORT and PRINT_IT procedures a pointer is used to simplify the coding and - since this is Pascal - making it easier to follow.

4) One item which will repay study and which should be tucked away for future use, is the 'unitread' at the beginning of the PRINT_IT procedure. It employs a pointer to designate the array and it demonstrates that a type identifier is a valid parameter for the SIZEOF function.

In operation the program prompts you for the unit number of the disk drive which has the disk you want to see the directory of. If you are like me and switch constantly between CP/M, Apple DOS and Pascal it takes some effort to remember that unit 11 is the 3rd disk drive under UCSD Pascal. So, the program provides you with all those equivalences.

After you give the unit number the program rewards you with a number alphabetically-sorted listing of the files on the disk similar to 'E(xtended directory list' under 'F(iler'.

When you are through, the program exits 'gracefully' to the operating system.

The program is easily convertible into a procedure or unit. It is one of my favorite utilities and I include it in most programs that I write.

If you don't want to type the program in yourself, send a program (or two?) or you own on disk to me and get this program on disk in exchange.

Randy Gresham : Legal Counsel
Saudi Arabian Parsons
PO Box 167
Yanbu al Bahr
Saudi Arabia

```
PROGRAM LIST_DIRECTORY;
        { This program reads the directory of a Pascal disk from within }
        { a program. Apple-Dayton Newuletter, Dec 81, John Matthews, MD }
        {    Totally overhauled by C. Brandon Gresham, Jr., 31 Dec 81   }

        {      Constants are set for APPLE ][+ running UCSD Pascal      }
        {                  using 5 1/4" disk drives                     }

const   MAX_DIR    =  77;          { Maximum number of entries in a directory.}
        MAX_UNIT   =  12;          { Maximum number of "units" }
        V_ID_LENG  =   7;          { Number of characters in volume ID. }
        T_ID_LENG  =  15;          { Number of characters in title ID.  }
        F_BLK_SIZE= 512;           { Standard disk block length}
        DIR_BLK    =   2;          { Disk address of directory }
```

```
type    DATE = packed record
                 MONTH : 0..12;          { 0 implies date not meaningful }
                 DAY   : 1..31;
                 YEAR  : 78..100         { 100 is temporary disk flag }
               end;

        DIR_RANGE = 0..MAX_DIR; { Number of entries (files) in directory }

        FILE_KIND = (UNTYPED,XDISK,CODE,TEXT,INFO,DATA,GRAF,FOTO,SECUREDIR);

        DIR_ENTRY = record
                      FIRST_BLK : integer;
                      LAST_BLK  : integer;
                      case DF_KIND: FILE_KIND of
                        UNTYPED,SECUREDIR : (VOLUME      : string[V_ID_LENG];
                                             BLK_COUNT,
                                             REC_COUNT,
                                             ZERO_BLK    : integer;
                                             LAST_BOOT   : DATE);
                        XDISK,CODE,TEXT,INFO,
                        DATA,GRAF,FOTO:     (TITLE       : string[T_ID_LENG];
                                             LAST_BYTE   : 1..F_BLK_SIZE;
                                             ACCESS      : DATE)
                      end;
        DIRECTORY = array[DIR_RANGE] of DIR_ENTRY;

var     I     : integer;
        DIR   : ^DIRECTORY;
        COUNT: DIR_RANGE;
        HEAP : ^integer;
        UNIT_NUMBER : 1..MAX_UNIT;
        MONTHNAME    : array[0..12] of string[3];
        INDEX        : array[DIR_RANGE] of DIR_RANGE;
        CH,CH1       : char;
        FLAG         : boolean;

procedure GET_UNIT_NUMBER;
begin
  read(CH);
  if ord(CH) <> ord('1') then
      case CH of
        '4','5','9'  : begin
                         FLAG := true;
                         UNIT_NUMBER := ord(CH) - ord('0');
                       end;
      end; { case }
  if ord(CH) = ord('1') then { should use else instead here }
    begin
      read(CH);
      case CH of
        '0','1','2'  : begin
                         FLAG := true;
                         UNIT_NUMBER := ord(CH) - ord('0') + 10;
                       end;
      end; { case }
    end; { else }
end;{ GET_UNIT_NUMBER }
```

```
procedure INITIALIZE;
begin
  MONTHNAME[ 0] := 'N/A';      { 0 implies date not meaningful }
  MONTHNAME[ 1] := 'Jan';MONTHNAME[ 2] := 'Feb';MONTHNAME[ 3] := 'Mar';
  MONTHNAME[ 4] := 'Apr';MONTHNAME[ 5] := 'May';MONTHNAME[ 6] := 'Jun';
  MONTHNAME[ 7] := 'Jul';MONTHNAME[ 8] := 'Aug';MONTHNAME[ 9] := 'Sep';
  MONTHNAME[10] := 'Oct';MONTHNAME[11] := 'Nov';MONTHNAME[12] := 'Dec';
end;

procedure sort;
        { Shell sort directory indirectly via 'index' array }
var     JUMP,
        TEMP,
        M,N        : DIR_RANGE;
        ALL_DONE   : boolean;
begin
  for I := 1 to COUNT do INDEX[I] := I;
  JUMP := COUNT -1;
  while (JUMP>1) do
    begin
      JUMP := JUMP div 2;
      repeat
        ALL_DONE := true;         { Assume it is in order already }
        for M := 1 to (COUNT - JUMP) do
          begin
            N := M + JUMP;
            if (DIR^[INDEX[N]].TITLE < DIR^[INDEX[M]].TITLE) then
              begin { Entries out of order, swap indices }
                TEMP := INDEX[N];
                INDEX[N] := INDEX[M];
                INDEX[M] := TEMP;
                ALL_DONE := false  { Swap required }
              end; { if }
          end; { for }
      until ALL_DONE
  end; { while }
end;{ SORT }

procedure PRINT_IT;
begin
  page(output);
  unitread(UNIT_NUMBER,DIR^,sizeof(DIRECTORY),DIR_BLK);
                                        {N.B. A type identifier is an acceptable
                                           parameter for the SIZEOF function. }
  writeln('#',UNIT_NUMBER,' is ',DIR^[0].VOLUME,':');
  writeln;
  writeln('NR.                 START LENGTH    DATE        FILE TYPE');
  writeln('==                  =====  ====  =========      =========');
  COUNT := DIR^[0].REC_COUNT;
  SORT;
  for I := 1 to COUNT do
    begin
      with DIR^[INDEX[I]] do
        begin
          if (length(TITLE)>0) then
            begin
              write(I:2,'_',
                    TITLE,'___':18-length(TITLE),
                    FIRST_BLK: 3,'__',
                    (LAST_BLK-FIRST_BLK):3,'____',
                    ACCESS.DAY  :2,' ',
                    MONTHNAME[ACCESS.MONTH]:3,' ',
                    ACCESS.YEAR :2.'      ');
              case DF_KIND of
                XDISK: write('BAD BLOCK');
                CODE : write('Code file');
                TEXT : write('Text file');
                INFO : write('Info?????');
                DATA : write('Data file');
                GRAF : write('GRAF?????');
                FOTO : write('FOTO?????');
              end;{cases of file kind}
            writeln;
          end;{if..then}
        end;{with..do}
      end;{for..do}
      writeln;
      write('Hit <RETURN> to continue ');
      CH1 := 'a';
      while (CH1 <> ' ') do read(CH1);
    end;{ PRINT_IT}

    procedure SCREEN_PROMPT;
    begin
      repeat
        page(output);
        FLAG := false;
        gotoxy(10, 6);
        write(' For which volume do you want the directory ? ');
        gotoxy(14,10);
        write(' 4:   Boot disk drive (slot 6, drive 1)');
        gotoxy(14,12);
        write(' 5:    2nd disk drive (slot 6, drive 2)');
        gotoxy(14,14);
        write(' 9:    5th disk drive (slot 4, drive 1)');
        gotoxy(14,16);
        write('10:     6th disk drive (slot 4, drive 2)');
        gotoxy(14,18);
        write('11:     3rd disk drive (slot 5, drive 1)');
        gotoxy(14,20);
        write('12:     4th disk drive (slot 5, drive 2)');
        gotoxy(14,23);
        write('Type "E" to E(nd processing.');
        gotoxy(60,10);
        GET_UNIT_NUMBER;
        if CH = 'E' then FLAG := true;
    until FLAG
end;{ SCREEN_PROMPT }


begin   { PROGRAM LIST_DIRECTORY }
  mark(HEAP);
  new(DIR);
  INITIALIZE;
  CH := '0';
  repeat
    SCREEN_PROMPT;
    if (CH <> 'E') then PRINT_IT;
  until (CH = 'E');
  page(output);
  release(HEAP);
end. {PROGRAM}
```

Visicalc Corner

by Frances Teo

This is the first of what I hope will be a regular corner with tips on using Visicalc. Whether or not it will be regular depends on you, the reader. I can waffle on for hours telling you of traps, etc, that I have fallen into but I want to know your problems so that I can help you in a positive way, so please send your letters post haste.

One of the most frequent complaints I hear is that Visicalc cannot print the formulae that you have in your model. However, if you have yards of paper and some spare time you can print everything that is in your model. /SS,S1 will send your printer into a frenzy and you off to make a cup of coffee. All labels and formulae with their grid references starting from the bottom right hand co-ordinate will be on your yards of paper (yards because Visicalc prints the content of each cell on a new line).

DIF files are the ideal way to tie separate models together when you do not have enough memory to do everything you want in one file. Arrange the information that is to be transferred in exactly the way it is to appear on the other model. With the cursor at the top left hand co-ordinate of the section to be transferred press /S£S. You are then prompted for the bottom right hand co-ordinate of the section in question. After this you are asked whether you want to save it by row or column, this should be obvious if you look at what you are transferring. When you give a file name it is useful to include the letters DIF at the end of the name so that you do not confuse it with a standard Visicalc file. After Visicalc has saved the file clear the screen and load in the model to which the information is being transferred. Place the cursor in the top right hand co-ordinate of the section where you want the information placed, press /S L, give the file name and state whether you want the file loaded by row or column. In a few seconds the information will be in the new model and you are ready to continue working. One of the many uses for DIF files is to transfer monthly balances through to the next month so that you may have year to date balances.

In the next edition of Hard Core we shall be discussing how to plan and construct a model.

One last thought, I had a phone call recently from someone in need of advice. She found it was very time-consuming redoing her model each time she wanted to use it. When you have finished creating a model save it as a Master copy and when you have entered data in it save it under a totally different file name. This means that you always have a fresh copy of your model ready for use.

AUTO WARNING

The short article taken from the IAC and reprinted in HARD CORE, August 1982, which describes a modification to add Auto Repeat on your keys gives a warning that it voids your warranty note. For our part we have to add that it could void your Apple, so don't do it! Apparently it can damage your encoder card.

LOOK! . . . A NEW NAME

apple ACCESSORIES

- All items fully guaranteed for one year
- Free postage and packing on all orders
- Immediat
- Many mo

BUSINESS SOFTWARE		INC. VAT
Calcstar (CP/M) (80 column spread sheet)	59.00	67.85
dBase II (CP/M)	349.00	401.35
Data Factory 5.0 (many new features)	157.00	180.55
Data Star (CP/M)	122.00	140.30
D.B. Master (version 3.02)	119.00	136.85
D.B. Master statistics	58.00	66.70
D.B. Master utilities (links with visi's)	58.00	66.70
Desk Top Plan II	105.00	120.75
Graphmagic (bar graphs, pie charts, etc)	47.00	54.05
Mathemagic	47.00	54.05
PFS Filing system (new improved)	68.00	78.20
PFS Graph	68.00	78.20
PFS Report	53.00	60.95
Supersort (CP/M)	75.00	86.25
Versaform (form generator in Pascal)	199.00	228.85
Visicalc (new version)	105.00	120.75
Visicalc expander (use with 32K/128K cards)	53.00	60.95
Visicalc utilities	34.00	39.10
Visidex	122.00	140.30
Visifile	132.00	151.80
Visipac (calc/trend/file)	378.00	434.70
Visiplot	99.00	113.85
Visischedule	155.00	178.25
Visiterm	59.00	67.85
Visitrend/plot	155.00	178.25
WORD PROCESSING		
Apple Speller (very fast proof reading)	45.00	51.75
Apple Writer II	72.00	82.80
Form Letter (use with Super Text II)	58.00	66.70
Format 80	275.00	316.25
Letter Perfect	87.00	100.05
Mailmerge (CP/M)	55.00	63.25
Pie Writer (40/80 columns)	78.00	89.70
Screenwriter II (70 col w/out 80 col card)	62.50	71.88
Super Text II	85.00	97.75
Wordstar 3.01 (CP/M) (enhanced features)	135.00	155.25
Zardax	139.00	159.85
GAMES		
Beer Run (Sirius)	16.95	19.49
Cannonball Blitz (Online Systems)	16.95	19.49
Choplifter (Broderbund)	16.95	19.49
Epoch (Sirius)	16.95	19.49
Flight Simulator (Sublogic)	16.95	19.49
Knights of Diamond (use with Wizardry)	18.95	21.79
Laff Pak (Online Systems)	16.95	19.49
Minotaur (Sirius)	16.95	19.49
Pinball (Sublogic)	16.95	19.49
Pool 1.5 (Innovative design	16.95	19.49
Swashbuckler (Datamost)	16.95	19.49
Tawala's Last Redoubt (broderbund)	16.95	19.49
The Prisoner (Edu Ware)	14.95	17.19
Threshold (Online Systems)	16.95	19.49
Time Zone (Online Systems)	43.75	50.31
Wizardry (Sir Tech Software)	24.95	28.69
Zork II (Infocom)	21.95	25.24

PRINTERS		
Diablo 630 (R/O)	1295.00	1489.25
Epson MX80T-3 (up to 132 col & hi-res)	299.00	343.85
Epson MX80FT-3 (as above & friction drive)	324.00	372.60
Epson MX82FT (very hi res graphics)	359.00	412.85
Epson Mx100FT 3 (100 cps & wide carriage)	429.00	493.35
Mannesman Tally MT 120 (160 cps)	365.00	419.75
Nec 8023 (100 cps, 136 col & prop. spacing)	319.00	366.85
Silentype & interface	159.00	182.85
Corona TPI (at last a low priced daisywheel)	429.00	493.35
Tec Starwriter FP1500 (25 cps)	650.00	747.50
PRINTER INTERFACE CARDS		
CPS Multifunction Card (inc real time clock)	119.00	136.85
Digitek Printmaster (BASIC/CPM/PASCAL)	69.00	79.35
Digitek RS232 (BASIC/CPM/PASCAL)	70.00	80.50
Grappler graphics (Epson/Anadex/Cen/P.tig)	98.00	112.70
MBP 16K (Epson 16K buffer)	96.00	110.40
Wizard 16K Buffer & graphics interface	149.00	171.35
80 COLUMN CARDS		
Smarterm (very cool running, many features)	182.00	209.30
Super-R-Terminal	182.00	209.30
U-Term (inc shift mod. & font editor)	129.00	148.35
Videx Enhancer II	83.00	95.45
Videx Softswitch (40/80)	19.95	22.94
Videx Utility Disc (inc font editor etc)	24.45	28.12
Videx Videoterm	163.00	187.45
Visicalc preboot disc (80 col with videx)	32.00	36.80
MONITORS/COLOUR CARDS		
Digitek Colour Card (excellent colour on TV)	87.00	100.05
Microvitec colour monitor	264.00	303.60
Microvitec colour card (use with above)	35.00	40.25
Sanyo 12" Green Screen	105.00	120.75
Zenith 12" Green Screen (very good value)	89.00	102.35
GRAPHIC UTILITIES & MUSIC		
3-D Supergraphics	23.00	26.45
Artist Designer	33.00	37.95
Digisolve Card (512 x 512 & 64k Ram)	375.00	431.25
E-Z Draw 3.3 (excellent graphic utility)	25.95	29.84
Graforth (fast 3D utility plus music)	43.00	49.45
Graphic package Sublogic (detailed 3D pack)	64.45	74.12
Higher Text II (many diff fonts, sizes, cols)	21.95	25.24
Pascal Animation	34.00	39.10
Pilot Animation Tools	34.00	39.10
Versawriter (graphic digitiser)	165.00	189.75
Versawriter Expansion software	23.00	26.45
Zoom Grafix (zoom in on hi-res screen)	23.95	27.54
All Music System (9 voice)	85.00	97.75
Echo II Speech Synthesiser	125.00	143.75
Electric Duet (creates 2 part music)	18.95	21.79
Music Machine 9 Voice (Vista)	79.00	90.85
Music System (Mountain Computer 16 voice)	211.00	242.65

E IN PACE

e delivery
re items available

☎ 0274 575973

Zapple Sound Effects & Music Board	56.00	64.40
UTILITIES		
Ace (Applesoft Command Editor)	21.95	25.24
Aplus (Applesoft structured Basic)	18.95	21.79
Assembly lang. Dev. (6502, Z80 or 8080)	75.00	86.25
Bag of Tricks	20.45	23.52
Back It up II+ (bit copier)	48.00	55.20
B.E.S.T. (Enhanced Software Tool)	23.95	27.54
Build Using (Provides 'print using' command)	23.95	27.54
Dakin 5 (12 utility programs)	47.00	54.05
Disc Library (inc. CP/M & Pascal)	33.95	39.04
Dos 3.3 Tool Kit	39.00	44.85
Edit Soft (powerful macro line editor)	18.95	21.79
Fast DOS	19.45	22.37
Global Program Line Editor (supports 80 col)	34.50	39.68
Image Printer Epson (flexible hi-res dump)	32.00	36.80
Lisa (Assembly lang. dev. system)	55.00	63.25
Lisa (Educational system)	70.00	80.50
List Master (inc. smart renumbering)	23.95	27.54
Locksmith 4.1 (bit copier for most discs)	61.00	70.15
Merlin Macro Assembler (editor & utilities)	42.00	48.30
Printographer (supports almost any printer)	27.95	32.14
Quickloader	15.95	18.34
Speed Star (compiles 1200 lines per min)	75.00	86.25
Super Disc Copy III	20.00	23.00
Super Kram	87.00	100.06
Tasc Compiler (handles very large programs)	95.00	109.25
The Inspector (disc snooper, needs 16k card).	35.00	40.25
The Routine Machine (mach. lang. routines)	34.50	39.68
SYSTEM EXPANSION		
16K Ramcard Digitek (with dos mover)	75.00	86.25
16K Ramcard Ramex	65.00	74.75
16k Ramcard Sanram	55.00	63.25
32K Ramcard Saturn	129.00	148.35
128K Ramcard (Saturn)	299.00	343.85
128K Ramex card	245.00	281.75
Clip on fan (inc separate power switch)	44.00	50.60
Cool stack (holds 2 drives & monitor + fan)	65.00	74.75
DOS upgrade kit (3.2 to 3.3)	36.00	41.40
E-Z Port (game socket extender)	11.45	13.17
Keyplus Numeric Keypad	83.00	95.45
Romplus Card in K/B Filter	119.00	136.85
The Mill 6809 with Pascal speed up	189.00	217.35
VC-Expand-80 (up to 14.5K Visicalc & 80 col)	65.00	74.75
VC-Expand Ramex (loads 136k visi in 20 sec)	40.00	46.00
Z80 card U-micro (card only)	73.00	83.95
Z80 card Microsoft (CP/M (Mbasic)	179.00	205.85
Z card (inc CP/M & Supercalc) very good value	179.00	205.85
LANGUAGES		
Apple Logo (available at last)	115.00	132.25
Apple Pascal	145.00	166.75
Apple Pilot	79.00	90.85

Cobol 80 (CP/M)	375.00	431.25
Forth II (Integer only)	39.00	44.85
Fortran 80 (CP/M)	103.00	118.45
Transforth (full floating point Forth)	68.00	78.20

Please make your order by completing the coupon and returning to us – or you can phone **DAVE or SHERIDAN**

0274 575973

and your enquiry will receive a friendly and speedy response

David *Sheridan*

PACE-SOFTWARE-SUPPLIES

Rose Bank, 130 Clayton Road,
Bradford BD7 2LY, West Yorks.

Please rush me the following items.

		£	p
1			
2			
3			
4			
5			
	TOTAL		

I enclose my cheque made payable to PACE SOFTWARE
(Export licence arranged)

Name ____________________

Address ____________________

Town ____________________

County ____________________

Postcode ____________ Tel. No. ________

HC 2/5

DOUBLEDOS REVISITED

By Colin Richardson

John Sharp's DOUBLEDOS article in the October, 1981, issue of Hard Core demonstrated a way of changing between DOS 3.2 and 3.3 without having to re-boot each time. However, it does have the slight drawback that it requires over 4.5K of memory. The method which will be described below reduces the amount of memory used to 2.25K plus about 160 bytes in page three. It also allows FID to be used on both DOS 3.2 and 3.3 disks, so you can copy programs from one format to the other without having to use MUFFIN or DEMUFFIN.

The original DOUBLEDOS program maintains copies of the RWTS routines for both DOS 3.2 and 3.3 in addition to the RWTS which is currently being used by DOS, and copies the appropriate one to DOS when an & command is issued. However, it is only really necessary to keep a copy of the RWTS routines which are not currently in use in addition to the ones actually being used, and this is where DDOS makes its saving in memory. The method to be described assumes a 48K machine.

INSTALLING DDOS

The first thing to do is to get a copy of the DOS 3.2 RWTS routines onto a DOS 3.3 disk, and this is done as follows. Boot up the DOS 3.3 System Master disk, BRUN BOOT13 and boot a 13 sector disk (e.g. the BASUG Introductory Disk). Enter the monitor with a CALL -151 and move the RWTS routines down to $900 as follows :

```
900<B700.BFFFM
```

Save these routines on a DOS 3.2 disk by typing

```
BSAVE RWTS 3.2,A$900,L$900
```

Now re-boot with a DOS 3.3 disk and MUFFIN RWTS 3.2 across to the DOS 3.3 disk.

The control program, which changes DOS versions, is shown in listing 1. This program runs at $300 but is loaded higher up in memory along with the RWTS routines and a loader program. Enter the monitor again and type in the hex code from $300 to $39D. Also type in the hex code shown in listing 2, from $803 to $839. Now move the control program up to $83A by using

```
83A<300.39DM
```

Finally load the DOS 3.2 RWTS routines from the DOS 3.3 disk :

```
BLOAD RWTS 3.2
```

and save the complete program with

```
BSAVE DDOS,A$803,L$9FE
```

USING DDOS

To use DDOS, just type BRUN DDOS. The RWTS routines will be stored above the DOS buffers out of harm's way, and the control program is stored in page three. &2 will now select DOS 3.2 and &3 will select DOD 3.3. If the & vector becomes disconnected, a CALL 768 (or 300G from the monitor) will reconnect it.

The alternative way of changing DOS versions is by the use of CTRL-V. A CALL 862 (or 35EG from the monitor) will alter the I/O hooks so that a CTRL-V changes from one version of DOS to the other. The current version is displayed in the top right hand corner of the screen. Each time a character is printed on the screen via the normal COUT routines, the DOS version is written to the top right hand corner and this means that the speed of any screen output is slightly reduced, but this is hardly noticeable. If this worries you, it is possible to alter the program so that the DOS version is only written each time a carriage return is printed. This DOS version indicator will appear to flicker as the screen display scrolls, but this is quite normal. When CTRL-V is in use, &2 and &3 will also cause the version indicator to change.

Since the use of CTRL-V requires the use of the I/O hooks and slows down the screen display, it is probably a good idea to use the & commands under normal circumstances. However, the main advantage of CTRL-V is that it allows FID to be used with both DOS versions. First BRUN DDOS, issue a CALL 862 to allow CTRL-V to be used, and then BRUN FID. All the options of FID work with both DOS versions, including Space On Disk, which gives the correct figures for used and free sectors on DOS 3.2 as well as 3.3

disks.

To use the Copy Files option to transfer programs between DOS versions, select the DOS version for the source disk and then select the FID Copy option. When asked to insert destination disk 'and press any key, press CTRL-V and this will change DOS versions - it will not be recognised as a normal keypress. Now press another key to continue the copy.

HOW IT WORKS

The method used is basically the same as that described by John Sharp in his original article. The loader program simply moves the RWTS routines up in memory from $900, where they are loaded, to $9400, and then moves the control program to page three. The DOS buffers are then moved down in memory using the method described on page 7-3 of "Beneath Apple DOS". This is done so that the RWTS routines are hidden away from DOS and any BASIC programs, and it means that they will not be erased by an FP or an INT.

The start of the control program sets up the ampersand vector to point at the COMMND routine which interprets & commands. SWAP is very much like the monitor's MOVE routine, but it swaps two ranges of memory over rather than simply copying a range of memory to a new area. It is used to exchange the RWTS routines when you change DOS versions. The IOSWAP routine beginning at $35E sets up the I/O vectors to divert input and output via NKEYIN and NCOUT respectively. NKEYIN looks for a CTRL-V and if it finds one it changes DOS versions. NCOUT first outputs the character in the normal way and then prints the DOS version number in the top right hand corner of the screen. Note that if you want to disconnect the CTRL-V option you should type IN£0<RETURN>, PR£0<RETURN>. Also note that to reclaim the memory used by the non-current RWTS routines you should re-boot the system.

Apologies to Richard Wilday whose nutty flowchart was printed in the last Hard Core without acknowledgement. It was Richard Wilday that did it, but he asked me not to bother mentioning it.

```
SOURCE FILE: DDOS
0035:              1 YSAV1    EQU   $35
0036:              2 CSWL     EQU   $36
003C:              3 A1L      EQU   $3C
0042:              4 A4L      EQU   $42
00B1:              5 CHRGET   EQU   $00B1
03EA:              6 REGSET   EQU   $03EA
03F5:              7 AMPRVEC  EQU   $03F5
9400:              8 RWTS2    EQU   $9400      ;NON-CURRENT RWTS
9CFF:              9 END      EQU   $9CFF      ;END OF THIS RWTS
B700:             10 RWTS     EQU   $B700      ;POSITION OF RWTS IN DOS
FCB4:             11 NXTA4    EQU   $FCB4
FD0C:             12 RDKEY    EQU   $FD0C
FD1B:             13 KEYIN    EQU   $FD1B
FDF0:             14 COUT1    EQU   $FDF0
0000:             15 *
0000:             16 *
------ NEXT OBJECT FILE NAME IS DDOS.BASUG
0300:             17          ORG   $0300
0300:             18          MSB   OFF        ;CHRGET GIVES HIGH BIT CLEAR
0300:             19 *
0300:A0 02        20          LDY   #$02
0302:B9 0C 03     21 SETVEC   LDA   JMPINST,Y
0305:99 F5 03     22          STA   AMPRVEC,Y
0308:88           23          DEY
0309:10 F7        24          BPL   SETVEC
030B:60           25          RTS              ;SET UP & VECTOR
030C:4C 0F 03     26 JMPINST  JMP   COMMND
030F:             27 *
030F:             28 *
030F:F0 44        29 COMMND   BEQ   EXIT       ;NO CHARACTER AFTER &
0311:48           30          PHA              ;SAVE CHARACTER
0312:20 B1 00     31          JSR   CHRGET     ;TIDY TXTPTR
0315:68           32          PLA              ;RESTORE CHARACTER
0316:C9 32        33          CMP   #'2        ;DOS 3.2 REQUIRED ?
0318:D0 30        34          BNE   CHECK3     ;NO
031A:AD 9C 03     35          LDA   TEXT1      ;WHICH DOS IN USE ?
031D:C9 B2        36          CMP   #$B2       ;  IS IT 3.2 ?
031F:F0 34        37          BEQ   EXIT       ;   YES, DON'T SWAP RWTS
0321:A0 08        38 SWAP     LDY   #$08
0323:B9 55 03     39 SWAP1    LDA   TBL-1,Y    ;SET UP A1-A4 FOR A MOVE
0326:99 3B 00     40          STA   A1L-1,Y
0329:88           41          DEY
032A:D0 F7        42          BNE   SWAP1      ;LEAVES Y=$00 ON EXIT
032C:B1 3C        43 SWAP2    LDA   (A1L),Y
032E:48           44          PHA
032F:B1 42        45          LDA   (A4L),Y
0331:91 3C        46          STA   (A1L),Y    ;SAVE PRESENT RWTS
0333:68           47          PLA
0334:91 42        48          STA   (A4L),Y    ;AND REPLACE IT WITH OTHER VERSION
0336:20 B4 FC     49          JSR   NXTA4
0339:90 F1        50          BCC   SWAP2
033B:AD 9D 03     51          LDA   TEXT2
033E:48           52          PHA
033F:AD 9C 03     53          LDA   TEXT1
0342:8D 9D 03     54          STA   TEXT2
0345:68           55          PLA
0346:8D 9C 03     56          STA   TEXT1      ;CHANGE VERSION INDICATOR
0349:60           57          RTS
034A:C9 33        58 CHECK3   CMP   #'3
034C:D0 07        59          BNE   EXIT       ;CHARACTER WAS NOT 3
034E:AD 9C 03     60          LDA   TEXT1      ;WHICH DOS IN USE ?
0351:C9 B3        61          CMP   #$B3       ;  IS IT 3.3 ?
```

```
0353:D0 CC                   BNE  SWAP         ;  NO, SO SWAP OVER RWTS
0355:60            EXIT      RTS
0356:00 94         TBL       DW   RWTS2        ;START OF RWTS TO BE MOVED
0358:FF 9C                   DW   END          ;END OF THIS RWTS
035A:00 94                   DW   RWTS2
035C:00 B7                   DW   RWTS         ;DESTINATION FOR RWTS2
035E:              *
035E:              *
035E:A0 03                   LDY  #$03
0360:B9 6C 03      IOSWAP    LDA  CSWTBL,Y
0363:99 36 00                STA  CSWL,Y
0366:88                      DEY
0367:10 F7                   BPL  IOSWAP       ;CHANGE I/O VECTORS
0369:4C EA 03                JMP  REGSET
036C:70 03         CSWTBL    DW   NCOUT        ;NEW COUT
036E:85 03                   DW   NKEYIN       ;NEW KEYIN
0370:              *
0370:20 F0 FD      NCOUT     JSR  COUT1        ;FIRST OUTPUT CHARACTER
0373:84 35                   STY  YSAV1        ;SAVE Y
0375:48                      PHA               ;  AND A-REG
0376:A0 06                   LDY  #$06
0378:B9 96 03      PRVER     LDA  TEXT,Y
037B:99 21 04                STA  $0421,Y
037E:88                      DEY
037F:10 F7                   BPL  PRVER        ;PRINT DOS VERSION
0381:68                      PLA
0382:A4 35                   LDY  YSAV1
0384:60                      RTS               ;EXIT WITH A,Y INTACT
0385:              *
0385:20 1B FD      NKEYIN    JSR  KEYIN
0388:C9 96                   CMP  #$96         ;CTRL-V ?
038A:F0 01                   BEQ  CHANGE       ;YES, CHANGE DOS VERSION
038C:60                      RTS               ;RETURN WITH CHAR IN A-REG
038D:20 21 03      CHANGE    JSR  SWAP         ;SWAP OVER RWTS
0390:8D 27 04                STA  $0427        ;UPDATE SCREEN
0393:4C 0C FD                JMP  RDKEY        ;NEED ANOTHER KEYPRESS
0396:              *
0396:                        MSB  ON
0396:C4 CF D3      TEXT      ASC  'DOS         3.
0399:A0 B3 AE
039C:B3            TEXT1     DFB  '3
039D:B2            TEXT2     DFB  '2

*** SUCCESSFUL ASSEMBLY: NO ERRORS
   3C A1L               42 A4L              03F5 AMPRVEC         038D CHANGE
 034A CHECK3            B1 CHRGET           030F COMMND          FDF0 COUT1
   36 CSWL            036C CSWTBL           9CFF END             0355 EXIT
 0360 IOSWAP          030C JMPINST          FD1B KEYIN           0370 NCOUT
 0385 NKEYIN          FCB4 NXTA4            0378 PRVER           FD0C RDKEY
 03EA REGSET          B700 RWTS             9400 RWTS2           0302 SETVEC
 0321 SWAP            0323 SWAP1            032C SWAP2           0356 TBL
 039C TEXT1           039D TEXT2            0396 TEXT              35 YSAV1
   35 YSAV1             36 CSWL               3C A1L               42 A4L
   B1 CHRGET          0302 SETVEC           030C JMPINST         030F COMMND
 0321 SWAP            0323 SWAP1            032C SWAP2           034A CHECK3
 0355 EXIT            0356 TBL              0360 IOSWAP          036C CSWTBL
 0370 NCOUT           0378 PRVER            0385 NKEYIN          038D CHANGE
 0396 TEXT            039C TEXT1            039D TEXT2           03EA REGSET
 03F5 AMPRVEC         9400 RWTS2            9CFF END             B700 RWTS
```

```
SOURCE FILE: DDOS LOADER
003C:                1 A1L      EQU  $3C
03D3:                2 COLDST   EQU  $03D3
FE2C:                3 MOVE     EQU  $FE2C
0000:                4 *
----- NEXT OBJECT FILE NAME IS DDOS LOADER.BASUG
0803:                5          ORG  $0803
0803:                6 *
0803:A0 08           7          LDY  #$08
0805:B9 29 08        8 MOVE1    LDA  MVTBL-1,Y
0808:99 3B 00        9          STA  A1L-1,Y
080B:88             10          DEY
080C:D0 F7          11          BNE  MOVE1       ;LEAVES Y=00 FOR MOVE
080E:20 2C FE       12          JSR  MOVE        ;MOVE RWTS UP TO $9400
0811:A0 08          13          LDY  #$08
0813:B9 31 08       14 MOVE2    LDA  MVTBL2-1,Y
0816:99 3B 00       15          STA  A1L-1,Y
0819:88             16          DEY
081A:D0 F7          17          BNE  MOVE2       ;LEAVES Y=00
081C:20 2C FE       18          JSR  MOVE
081F:20 00 03       19          JSR  $0300       ;SET UP & VECTOR
0822:A9 93          20          LDA  #$93
0824:8D 01 9D       21          STA  $9D01       ;MOVE DOS BUFFERS DOWN BELOW RWTS
0827:4C D3 03       22          JMP  COLDST      ;DOS COLD START
082A:00 09          23 MVTBL    DW   $0900       ;START OF RWTS
082C:FF 11          24          DW   $11FF       ;END OF RWTS
082E:00 09          25          DW   $0900       ;START OF RWTS
0830:00 94          26          DW   $9400       ;DESTINATION
0832:3A 08          27 MVTBL2   DW   $083A       ;START OF DOS CHANGE ROUTINES
0834:D7 08          28          DW   $08D7       ;END
0836:3A 08          29          DW   $083A       ;START
0838:00 03          30          DW   $0300       ;DESTINATION

*** SUCCESSFUL ASSEMBLY: NO ERRORS
   3C A1L           03D3 COLDST         FE2C MOVE          0805 MOVE1
 0813 MOVE2         082A MVTBL          0837 MVTBL7          3C A1L
 03D3 COLDST        0805 MOVE1          0813 MOVE2
 082A MVTBL         0832 MVTBL2         FE2C MOVE
```

(Editor: In case some of our newer Apple users don't know see what the previous article is all about let me just remind you of the problem. Very often you buy a new disk of powerful software only to discover that it won't boot on any machine you try. Panic, call up the supplier only to be told that there is no problem, all you have to do it boot the system using the BASICS disk that came with your system or BRUN BOOT13 on the System Master Disk. This will enable software created under the 'old DOS' - called 3.2 or 13 Sector - to run under the new DOS - 3.3 or 16 sector. Without going into any more technicalities, let me just say that the above procedure is easy but a bit of a bore, and this is why various methods have been devised to enable software to run whatever the format.)

Developer

The DEVELOPER is a powerful program development aid for programmers writing in APPLESOFT.

DEVELOPER speeds up the program development cycle by providing facilities to make rapid local and global changes to program text. DEVELOPER consists of 4 modules, each of which is accessed by typing the ampersan '&' followed by a single command letter and a line number or string of characters. DEVELOPER is designed to operate either on its own or in conjunction with the RENUMBER utility present on the DOS 3.3 SYSTEM MASTER or the APPLESOFT PROGRAMMER'S ASSISTANT supplied with the DOS 3.3 TOOL KIT. Once installed DEVELOPER is completely transparent and does not interfere with DOS or APPLESOFT.

The 4 modules that comprise DEVELOPER are:

* A Line Editor offering rapid cursor moves, insert and delete.

* A Global Editor allowing all lines containing a specified string to be listed and an optional string substituted for it.

* A Lister with forward and reverse list, single step and suspend list.

* A System Variable Dump displaying much useful information on memory status.

SYSTEM REQUIREMENTS DEVELOPER requires a 48K APPLE II+ with APPLESOFT in ROM or language card and at least one DISK II running under DOS 3.3.

£19.95 + VAT

lower case chip + SHIFT KEY MOD

- Adds the full lower case set to your APPLE II
- Complete with clip-on SHIFT KEY modification
- Lower case letters have true descenders
- Compatible with most word processing packages
- Available in most European languages
- Comprehensive documentation covers installation & software

```
 !"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
`abcdefghijklmnopqrstuvwxyz{|}~
```

Requires Revision 7 (and upward) APPLE II
PRICE £34.00 + V.A.T.

HELP FOR EPSON/APPLE OWNERS
LCC EPSON EPROM – by Mike Glover

If you have an EPSON graphics printer with the 8132 HIRES parallel card then this inexpensive enhancement is a must. The LCC Eprom has been designed to:–

- Support standard parallel printer card commands
- Enables all the HIRES options to be controlled using simple command characters. (No need to POKE)
- Turn VIDEO on and off without POKES
- Stop program crashes
- Complete with full documentation and programming examples to enable you to get the most from your EPSON.
- Add a new range of HIRES graphics set up commands to enable you to select screens, size, inverse etc. without a single POKE! **£18.00 + V.A.T.**

LEICESTER
computer centre limited

67 Regent Road, Leicester LE1 6YF. Tel: 0533 556268

BEGINNERS PAGES

STRINGS

Soon after starting to program in BASIC one realises there are two types of variables - numbers and 'strings' of characters. Numbers can be assigned to string variables, but they are as a set of characters just like words; and do not have any meaning as regards the position of the number in the sequence as occurs with numerical variables.

Applesoft has a number of commands to handle strings. Integer Basic does not have as many. There are a number of ways of getting round the problems. Some of these are described in the book APPLE II USERS' GUIDE, others are outlined and used in the program "THE INFINITE NUMBER OF MONKEYS" by Bruce Togazzini on the library disk 25.

In the following I do not propose to give the full command structure in each case. The syntax is in the APPLESOFT manual or books like "APPLE II USERS' GUIDE".

With these commands we can add strings together:-

```
10 A$ = "ABCD"
20 B$ = "EFGH"
30 C$ = A$ + B$
40 PRINT C$
```

and C$ will be printed as ABCDEFGH.

This process can also be useful in the addition of "illegal" characters to strings which have been input. APPLESOFT will remove leading spaces if you try to type them in, and in some cases you need them in another part of your program. This can be accomplished by the following:-

```
10 INPUT A$
20 B$= CHR$(32) + A$
30 C$= " " + A$
```

BOTH lines 20 and 30 undertake this manipulation in different ways. Other characters such as "," can be added in the same way.

We can remove them from the end of a string

```
10 A$ = "ABCD"
20 B$ = LEFT$ (A$,2)
30 C$ = RIGHT$(A$,3)
40 PRINT B$
50 PRINT C$
```

This will then print AB and then BCD, as the values of B$ and C$.

We can combine these operations to move one or more characters from one end of a word to another:-

```
10 P$ = "ABCD"
20 Q$ = LEFT$(A$,1)
30 R$ = P$ + Q$
40 S$ = RIGHT$(R$,4)
```

In practice this would be done in one expression

```
40 S$= RIGHT$(P$,3) + LEFT$(P$,1)
```

Normally, we would not necessarily know the length of the string, as it could vary, or be the result of an INPUT statement. There is another command to allow you to find the length of the string A$ - LEN(A$). Now we can shift from one end of any length string!-

```
10 INPUT X$
20 Y$ = LEFT$(X$,1)
30 Z$ = X$ + Y$
40 P = LEN(Z$) - 1
50 W$ = RIGHT$(Z$,P)
```

Where might one use this? One place would be the case as follows.
Suppose we had a list of names and we wished to sort them. We would normally do this on the basis of the first letter of the surname. This means we would store the surname as the first word followed by the forename. However, as we know from when BASUG's membership database was written like that, some people object when they receive letters addressed to BLOGGS FRED. One way to get around this is to switch the strings around. We then make use of what is known as a delimiter - in this case the space between the words, to know where to make the split. There are two parts to this. The first is to find the space and the second is to do the switch.

Finding the space is accomplished by means of the command MID$. In practice this is done as follows:-

```
10 INPUT A$
20 FOR N = 1 TO LEN(A$)
30 IF MID$(A$,N,1) = CHR$(32) THEN 100
40 NEXT N
50 PRINT "NO SPACE!!!" : END
100 B$= LEFT$(A$,N-1)
110 C$= RIGHT$(A$,LEN(A$)-N)
120 D$= C$ + " " + B$
130 PRINT D$
```

These various functions are vital for handling strings. Some BASICs contain another command called INSTR. This allows a target string to be searched for in another string. This is especially useful in the case of a question and answer system, where one wants to trace a particular answer in a host of variations e.g. the string "RABBIT" out of such possibilities as "A RABBIT" or "RABBIT" or "RABBITS". This is where APPLESOFT begins to look a bit long in the tooth. Perhaps someone would like to write in with a simulation of the INSTR function, from the string handling functions already present.

MID$ can be a very useful in another context. Sometimes it can be a problem storing many variables in arrays. The variables can be stored in strings. The months of the year for example:-

```
10 YEAR$ = "JANFEBMARAPRMAYJUNJULAUGSEPOCTNOV
DEC"

20 INPUT "WHICH MONTH DO YOU WANT
(1-12)" ; A

30 MONTH$ = MID$(YEAR$, ((A-1) + 1 ), 3)

40 PRINT "THE MONTH IS :- " ; MONTH$
```

This is only of use when the substrings are of the same length. There is a famous method of inserting machine code into memory by using this technique. Perhaps someone would care to write about the LAM routines for doing this.

YOUR EAMON PROBLEMS SOLVED

by John Martin

A number of queries have been received about EAMON disks, and we are now able to deal with the problems raised.

1. The Eamon Master disk (E1) is required not only to run " Beginner's Cave " (on the same disk) but also to run ANY other EAMON adventure (disks E2-E6)

2. Some copies of the Master Disk are now known to contain a faulty set-up program. You can ascertain whether this applies to your disk by LOADing and LISTing the program " The Wonderful World of EAMON ". If you spot nonsense within this program, return it for replacement.

3. Otherwise, proceed as follows :-

a) Remove write protect tab from disk E1

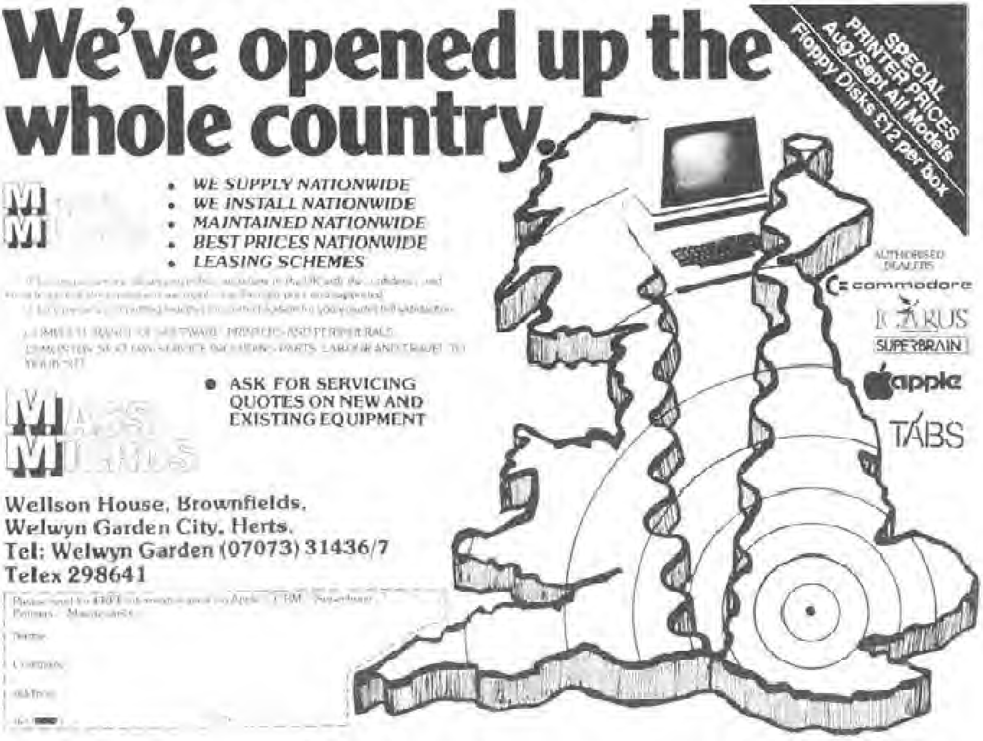
b) Run Master Disk program " Set up for 32K ", if you have 48K RAM.

c) RUN " The Wonderful World of EAMON "

d) Follow the instructions on the screen

e) When prompted, either press 'C' key for Beginner's Cave on the same disk, or replace with another adventure disk and THEN press 'C' key.

Happy Adventuring !!



EDUCATION COLUMN

THE MAGIC BLACKBOARD - REVIEW
by Norah Arnold

The Magic Blackboard is a 'join the dots' drawing program by Jonathan and Gillian Seagrave and is intended for use by physically handicapped children, mentally handicapped people of all ages and normal youngsters at pre-school and infant stage.

When the first menu comes up, a choice is given between displaying just pictures, just words or words and pictures together. Having made your choice, you are then asked if you want the standard display. If you reject the standard display you are asked to make a series of choices. The first of these asks whether the picture is to be drawn using one paddle button or two paddle buttons. If the second is selected then the button on each paddle must be pushed alternately. This would seem a sensible option for a parent using the program with a very young child or for two children taking turns.

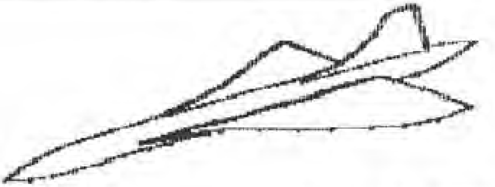
Next you are asked if you want sound. If you answer yes, then the drawing of the picture is accompanied by sound which is used in a manner which tends to add a little excitement in the eyes, or ears, of small children. Each press of the paddle button causes a note to be played in a rising chromatic scale.

Colour choices come next; green, magenta, white blue and red are available. The next option is for fast or slow plotting of the dots, followed by a choice of the thickness of the line, from 1 to 9.

Having completed these choices and also chosen which disk to use, four cartoons of animals are displayed on the screen. One of these four, rabbit, kangaroo, tortoise or butterfly, can be used to draw the picture. The chosen animal moves from dot to dot pointing to the dots with part of its body.

One can then choose to see the list of pictures or words available, before typing in the name of the chosen item. After a short wait the dots for the picture are displayed accompanied by the chosen animal 'pointing' to the first dot. Pressing the paddle button causes one line to be drawn in and the first note to be played.

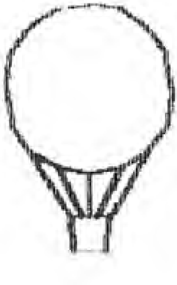
When the picture or word is completed, pressing ESC brings up another menu. Lines can be added to the picture to individualise it by fixing points and then joining them with a line. Any of the four animal cartoons can be placed on the picture. The final picture can be printed on a Silentype or a Trendcom printer. This ability to produce the drawing on paper would be very important to physically handicapped children who cannot produce a drawing with pencil and paper themselves.

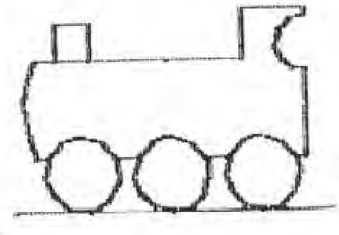


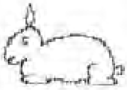
The final menu lets you have another go with the same choices, or with new choices or see your last picture again.

Although this program has a very structured approach and the role of the child is relatively passive, I think it has a definite place for use by those categories of people mentioned in the first paragraph. If joining the dots round words such as LADIES, GENTS, BUS STOP and DANGER will help mentally handicapped people cope with everyday situations then it has made a significant contribution.

I thought a faster loading routine wouldn't come amiss; some of the waits were a little long.







readers' letters

Nuneaton

Dear Sir...

I have a 48k machine with cassette. My biggest problem is that nearly all the commercial software is only available on disk and is often protected. Despite this I have managed to convert a large variety of software that does not require disc access while running. However, I have been unable to find an Apple assembler (not a mini-assembler type) which will assemble source from memory into memory (and therefore could be made compatible with a cassette system). Since I now write mostly in machine code I find this a distinct disadvantage when trying to write large (>1k) machine code programs and especially when trying to alter small portions of a program. The result is often sloppy and not completely efficient programs. I have heard that there is a cassette version of the LISA assembler for Apple but have been unable to find a stockist. I would be grateful for any solution to these problems.

Yours hopefully

Andrew Beveridge

Dear Hardcore,

In the last issue of Hardcore, there was a cry for help from a 'computer widow'. Many of the things said struck a chord, and to cut a long story short, I discovered that it was my wife who had written it. Those who have read the article will note that towards the end of the article, she said that she would rather compete with another woman. Who are we mere men to argue?

Taking her at her word, I have now found a new partner, and have left the family home. LET THIS BE A LESSON TO ALL WINGEING WIVES.

Yours etc.

Verbatim® Datalife™

disks for the performance of a lifetime

NOW disks, with the five year warranty

MINIDISKS

Verbatim, the world's leading producer of minidisks, makes the widest range of minidisks available for word processing and data processing applications. Intent on maintaining this position of leadership, we constantly strive to make our minidisks even better.

Each Verbatim Datalife™ minidisk is equipped with a hub reinforcing ring to aid in registration, reduce errors and give better alignment repeatability. Our dual-sided, double density Datalife™ minidisks offer users higher storage capacity. Our new plastic box makes our minidisks easier to use, safer and more convenient to store. Our testing standards go beyond the industry standard because we insist on Verbatim being the industry standard of excellence.

Introducing

CLEANING DISKS

The Verbatim Datalife™ Head Cleaning Disk is a convenient, reliable, safe means of removing up to 90 percent of debris contaminating magnetic recording heads used in computer and word processing systems.

The Verbatim Datalife™ Head Cleaning Kit consists of a durable, reusable, Lexan® jacket and presaturated, disposable cleaning disks.

The Verbatim approach is superior to and less expensive than other cleaning disks presently available.

The kit, available in 8" and 5¼" sizes, differs from those already available in the marketplace in the following ways:

- The Verbatim cleaning disks are disposable. This approach allows a fresh clean disk to be used each time. No accumulated debris from previous cleanings comes in contact with the heads.
- Each cleaning disk is presaturated with a precise amount of cleaning solution. Presaturation eliminates the inconvenience associated with splashing solvents onto a disk.

Head Cleaning kits 5 1/4 in only
#6 incl P&P
Replacement cleaning disks
#11 per box of 10 incl P&P
Available from BASUG

SPECIAL OFFER TO B.A.S.U.G. MEMBERS ONLY

10 x MINIDISKETTES **£19·50** INCLUSIVE OF V.A.T. AND POST.
I ENCLOSE A CHEQUE MADE PAYABLE TO B.A.S.U.G. CHEQUE No .... PLEASE FORWARD TO ME BY RETURN AT THE ENCLOSED ADDRESS.

NAME ....

ADDRESS ....

.... POST CODE

London N.W.3.

Dear David,

Graham Rubens (Letters - June 1982) asks for a DOS patch to stop DOS from opening an empty text-file when a named file does not already exist on disk.

May I suggest an easier method which does not involve any DOS patching. Enable error trapping then VERIFY the file. If it doesn't exist, the error trap will be "sprung" with the "FILE NOT FOUND" error and can be used to skip the unwanted "OPEN" command. Here is an example :-

```
.
.
.
100 ON ERROR GOTO 1000
110 PRINT D$ "VERIFY TEST.DATA"
120 POKE 216,0
130 PRINT D$ "OPEN TEST.DATA"
.
.
.
1000 POKE 216,0
1010 IF PEEK (222) = 6 THEN PRINT "FILE NOT ON
DISK"
.
.
```

Graham also asks why Apple Writer cannot cope with two disk controllers. The answer is that Apple Writer's internal commands for transferring between its two modules are "BRUN TEDITOR,D1" and "BRUN PRINTER,D1". "D" ($C4) can be can changed to "S" ($D3) at $12CB (in Teditor) and $102C (in Printer).

The third problem, which I can only answer here in outline, is how to put Teditor and Printer into memory at the same time without using a Ramcard. Each module starts at $803 and ends just before $1900, so taking up roughly $1100 bytes (i.e. 3.75K each). The text buffer is located between $1900 and DOS's buffers at $9600, taking up approximately $7C00 bytes - about 19K.

Two ways of putting both modules into memory are a) to store them both under DOS and then copy the appropriate one down to $803 as required and b) to relocate the second module at the top of memory and call it there.

The first method, which follows the Ramcard idea, needs twice 3.75K and would reduce the text buffer to about 11.5K. The second way needs only 3.75K, leaving 15.25K. Extra memory could be released by reducing MAXFILES to 1.

Unfortunately, Apple Writer does not make automatic internal adjustments to compensate for changes in HIMEM and MAXFILES and these would need to be altered manually in the code. Furthermore, the second method would involve off-setting all absolute addresses in the code so that the re-located module would run properly at its new location - not too difficult by using a symbolic disassembler (such as SYMDIS) and then re-assembling the source code.

If there is sufficient interest from BASUG members (letters to Hard Core and not to me please), I will try to work out the methods in detail. However, it should be mentioned at this stage that Apple Writer is a copyrighted program and the source code of the patched programs cannot be published or put into the BASUG library.

Ian Trackman

Membership Matters

First let me wish you all a Very Merry Christmas and a Happy New Year. As the year ends my work really starts and with some luck I will be able to process all the renewals before the next Hard Core publication date, February. To do this I need your help - if members send their renewals in early there will be ample time. You will find a renewal form in this month's Hard Core. I would ask you to use this to renew, since it speeds the process up enormously. Please use block capitals and include your telephone number and post code.

At the bottom of the form you will see a declaration to be signed by members. This declaration has been printed on the advice of BASUG's legal adviser and is required if we are to be a limited liability organization.

The wording means that if we do become a limited liability company all the members will be liable to pay a MAXIMUM sum of ONE POUND in the event that the club should ever close leaving debts behind. This liability continues in force for one year after members leave BASUG. This type of liability by guarantee means that no directors are needed or appointed, that the club continues to be run by the committee, as at present. One benefit is that we shall be forced to keep proper books under the Companies Act.

If any members are unsure about this declaration please write to me and I will try to explain it more clearly.

Now on to the more mundane tasks. Some members of the club need to be reminded that that we are NOT a commercial concern, that is to say, 90% of the work done for members is by voluntary help, which means that, as we have said in the past, members of the committee and other volunteers work until the small hours getting things done. If this were done on a commercial basis your software disks would be about £10 and we would no longer find it possible to make products available to you at just a little over cost.

I mention this because in the recent past I have received a number of querulous letters demanding to know why membership packs have not been dispatched within ten days or why software disks have taken two weeks to arrive. You have to understand that members doing this work do not live inside PO Box 174, Watford, and come from far afield, often travelling 70 miles to get there. We all have work to do, not to speak of leading a social life.

I would therefore ask members to bear this in mind and not forget that many mail order companies insist on allowing 28 days for delivery or reply. Having said that, if you do have any legitimate cause for complaint please have no hesitation in writing to me.

local groups

Hants and Berks Local Group

Fran Teo has decided it is high time something is done for the deprived Apple users in the Hampshire Berkshire area. You need a local group of BASUG, she tells me. So if you are interested in attending a first meeting which will deliberate on what is to be done, call Francis Teo on Bracknell [redacted] and talk about it.

Northwest Frontier

A.K. Mitra has asked to announce that he would like to explore the possibilities of setting up a BASUG local group in the N.Staffs/Cheshire/Manchester area. His telephone number is Congleton [redacted]

If you are forming a local group you can get assistance from our membership secretary, Jim Panks. He can help publicise your efforts by contacting BASUG members in your area. So please, write to Jim Panks at our Watford Box No and mark it "Local Group Coordinator".

Accutrack Disks . . .

Because data reliability is the important difference in disk construction.

Anatomy of a disk.

Flexible disks are simple information storage devices consisting of a magnetic disk enclosed in a semi-stiff protective jacket. The disk rotates within the jacket while magnetic recording heads on your data or word processing systems "read" or "write" information on the disk's magnetic surface. Since disk operation is simple, it's relatively easy to make one that works. But building in reliability is something else again. It takes specialized technology to build disks that operate flawlessly over an extended period of time.

What counts in disk construction.

Key design objectives for a disk are listed below. How well a disk measures up to these objectives relates directly to the throughput, accuracy and overall costs for your data or word processing system. No disk measures up better than Accutrack.

- The magnetic coating must be precisely formulated and uniformly applied. Imperfections as small as five millionths of an inch cause signal dropouts, data checks and wasted processing time as well as errors.
- The disk surface must be absolutely clean, totally flat and permanently lubricated to prevent excessive head wear with subsequent signal degradation and eventual loss of information. (This is the most critical objective and the one that's most often compromised since poor operating results take a while to show up. It's also the area that most effects the long term reliability of your data.)
- The disk must be free to rotate within its jacket without internal drag to avoid further data checks, excessive processing times and errors.
- The jacket must protect the disk from external contamination and damage. It should also remove microscopic particles of debris from the disk surface before they can damage the disk.

Why you'll never find the best disk bargain in the bargain basement.

While there's little apparent difference between other disks and Accutrak, the performance differences can be substantial. Simply stated, an Accutrack disk is premium priced. But the protection it gives your information, the reliability it provides to your operations, and its substantially longer life make it the best disk buy. After all, the real cost of your operations is constructing and processing the data stored on the disk – not the disk itself. It doesn't make sense to trust that data to anything but the best disk. Accutrack.

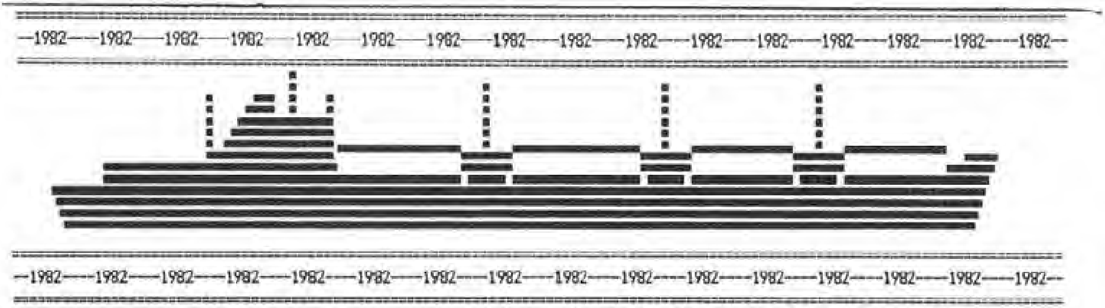
BASUG BULK BUYING — 500off PRICE FOR 10s

£17:50 incl FREE LIBRARY CASE !

(ADD 50p POSTAGE / PACKING)

BRITISH APPLE SYSTEMS USER GROUP

P.O. Box 174 Watford WD2 6NF



```
=========================================================================================
--1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982--
=========================================================================================
```

```
=========================================================================================
--1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982----1982--
=========================================================================================
```

by Peter Trinder

The calendar in this month's Hard Core was produced by utilising several routines from different sources. I must first thank Max Parrott whose main calendar program forms the foundation and does the calculating of the actual month/date part. This program was published in the December, 1981 Windfall.

I have replaced his header with a graphic picture which has been incorporated into the program using routines supplied in the 'Printographer' a Hi-res Graphics printing package from Southwestern Data Systems and sold here by SBD.

The Printographer allows you to load and 'edit' by cropping a Hires image both horizontally and vertically. There is a clever means of saving a Hires image in less than 34 sectors thereby saving disk space. The disk includes a programme which enables the Hires image to be included in a basic program. At the same time the Hires screen can be cropped top and bottom and 'tabbed' into position.

There is one area in which the Printographer excels. It contains printer drivers for all the known printers made and a configuration program to match these printers to most of the usual printer cards including Apple Parallel - Apple Serial - Grappler - CPS Parallel & Serial and others.

SDS have provided facilities to make three backup copies which should satisfy most people's needs. This backup policy applies to most of their software and is a sensible compromise in my view.

Apart from the calendar printed here with the Apple logo I have produced one with my own company logo as per the enclosed sample. The logo was constructed using the HRCG on the DOS Toolkit.

```
0  REM ************************
1  REM   CALENDAR BY MAX PARROTT
2  REM    * WITH ENHANCEMENTS BY
3  REM     *    PETER TRINDER
4  REM
9  GOSUB 4000: REM    ***LOAD GRAP
     HICS***
10  DIM D(2,3),P(2,5,6),A$(6),M$(2
     ,3),N$(9,4,6)
20  HOME : INPUT "PLEASE ENTER THE
      YEAR REQUIRED ";Y2
30  IF Y2 < 1900 THEN Y2 = Y2 + 19
     00
40  FOR J = 0 TO 3: FOR I = 0 TO 2
     : READ D(I,J),M$(I,J): NEXT :
      NEXT
50  FOR I = 0 TO 6: READ A$(I): NEXT

60 D$ =  CHR$ (4):M2 = 1:M = 1
70 E$ =  CHR$ (1):N$ =  CHR$ (7):
200 D(1,0) = D(1,0) + (Y2 / 4 =  INT
     (Y2 / 4)):Y1 = 1903:M1 = 1:D1
      = 1:D2 = 1
210 A =  INT (365.25 * (Y1 - (M1 <
     3))) +  INT (30.6 * (M1 + 1 +
     (M1 < 3) * 12)) + D1
220 B =  INT (365.25 * (Y2 - (M2 <
     3))) +  INT (30.6 * (M2 + 1 +
     (M2 < 3) * 12)) + D1
230 D =  ABS (B - A):D = D - 7 *  INT
     (D / 7):D = D - 3: IF D < 0 THEN
     D = D + 7
235  HOME
240  PRINT "START DATE IS ";A$(D);
     ", ";M$(0,0);", ";" ";Y2
250  PRINT : PRINT : PRINT "CHECK
     THAT THE PRINTER IS ON"
260  PRINT "WHEN READY PRESS THE S
     PACE BAR"
270  GET I$
280  FOR I = 0 TO 9: FOR J = 0 TO
     4: FOR K = 0 TO 6
290  READ N:N$(I,J,K) =  CHR$ (N):
      NEXT : NEXT : NEXT
300  GOSUB 6000
306  FOR X = 1 TO 89: PRINT "=";: NEXT

307  PRINT
308  FOR X = 1 TO 11: PRINT "--"Y2
     "--";: NEXT : PRINT
309  FOR X = 1 TO 89: PRINT "=";: NEXT
```

```
310  PRINT
 1310  CALL 35456, HGR ,N,H,2,20,2
     3,80
1350  GOSUB 6000
1351  FOR X = 1 TO 89: PRINT "=";:
      NEXT : PRINT
 1352  FOR X = 1 TO 11: PRINT "--"
Y2"--";: NEXT : PRINT
1353  FOR X = 1 TO 89: PRINT "=";
: NEXT : PRINT
1410  FOR L = 0 TO 3: GOSUB 3000
1412  PRINT E$
1415  PRINT  SPC( 21);: FOR I = 0
 TO 2: PRINT M$(I,L); SPC( 2
1);: NEXT I
1417  PRINT N$
1420  FOR I = 0 TO 2:N = 1: FOR J
 = 0 TO 5: FOR K = 0 TO 6
1430  IF K <  > D THEN  NEXT K
1440 D = D + 1: IF D = 7 THEN D =
0
 1450  IF N > D(I,L) THEN D = K: GOTO
1480
 1460 P(I,J,K) = N:N = N + 1
1470  NEXT K: NEXT J:D = D + 1
1480  NEXT I
1500  FOR K = 0 TO 6: PRINT E$;A$
(K);N$; SPC( 7);: FOR I = 0 TO
2: FOR J = 0 TO 5
1510  IF P(I,J,K) < 1 THEN  PRINT
"-   ";: GOTO 1530
1520  PRINT P(I,J,K); SPC( 2 + (P
(I,J,K) < 10));
1530  NEXT J: PRINT "    ";
1540  NEXT I: PRINT
1550  NEXT K
1560  NEXT L
1900  FOR X = 1 TO 89: PRINT "=";
: NEXT : PRINT
1920  FOR I = 1 TO 11: PRINT "--"
Y2"--";: NEXT I: PRINT
1940  FOR X = 1 TO 89: PRINT "=";
: NEXT : PRINT
1990  PRINT : PRINT D$"PR£0"
2000  END
3000  FOR I = 0 TO 2: FOR J = 0 TO
5: FOR K = 0 TO 6
3010 P(I,J,K) = 0: NEXT : NEXT : NEXT
: RETURN
4000  HIMEM: 35456:D$ =  CHR$ (4)
: REM  CTRL-D
4010  PRINT D$;"BLOAD PRINT GRAPH
ICS,A$8C00"
4015  PRINT D$;"BLOAD PRINT GRAPH
ICS.O,A$8A80"
4020  PRINT D$;"BLOAD APPLE LOGO.
PIC,A$2000"
4030  RETURN
 5000  DATA 31,JAN,28,FEB,31,MAR,3
 0,APR,31,MAY,30,JUN,31,JUL,3
 1,AUG,30,SEP,31,OCT,30,NOV,3
 1,DEC
```

Chelsfield

Dear Sir,
On Page 41 of Hardcore Vol 2 N. 5 you refer to Multitech of Taiwan. I have telephoned the Taiwan Board of Trade for Multitech's address, but they told me that they had no such company listed. Can you please send me either a more complete name of Multitech or better still their contact address in Taiwan.

Yours
G Mack

(Ed. Can't imagine why the Taiwan people are being so coy about this. Multitech is a serious organization which manufactures among other things the reputable Micro-Professor (MPF-1), a Z80 based micro "that will lead you step-by-step to a thorough knowledge of a microprocessor". Their MPF-II on the other hand is a "full-feature" home computer which is "compatible" with Apple II but apparently their own product. The address, as published in Byte for November, is

Multitech Industrial Corporation
977 Min Shen E Road
Taipei
105 Taiwan R.O.C.

As soon as supplies come in the unit will be marketed by Flight Electronics Ltd, Flight House, Quayside Road, Brittain Manor, Southampton, Hants. But, dear reader, stay your pen: the price, the price: at £235 plus shipping plus VAT making £275.94 in all you might like to think it over.)

(Editor: In Peter Trinder's listing you will notice a rogue 'GOTO' and 'NEXT'. It happens to my listings a lot too, despite all my Poke 33,33's. These jaggedy bits are anathema to a knife-wielding editor, who loves to see things in straight lines. Dear Readers, how can I get my GOTOs back into line?)

HOW TO CALCULATE SAMPLE VARIANCE AND STANDARD DEVIATION ACCURATELY

By R.D. Purves

Sample variance and standard deviation are statistics which measure the extent to which individual observations are "scattered" about the arithmetic mean value. People wishing to calculate these elementary statistics often reach for an old-fashioned text-book in which they find instructions for the "desk calculator" method. In Applesoft, one might write:

```
100 SIGMX = 0.0:SSQ = 0.0:N = 0
110 PRINT "SAMPLE £" N + 1;
120 INPUT " <-9999 TO END> ? "; X
130 IF X = -9999 GOTO 160
140 SIGMX = SIGMX + X:SSQ = SSQ + X ^ 2
150 N = N + 1: GOTO 110
160 VAR = (SSQ - SIGMX ^ 2 / N)/(N - 1)
170 PRINT "MEAN = "SIGMX / N
180 PRINT "SAMPLE VARIANCE = "VAR
190 PRINT "STANDARD DEV. = " SQR (VAR)
```

This innocent looking program conceals some numerical horrors. It is algebraically correct (sample values 1.0, 2.0 & 3.0 give variance = 1.0), and does not make the mistake of forcing the user to count the number of data items to be entered (computers count better than people do). But it gives poor accuracy for tightly bunched data. Sample values 2000.0, 2000.1, 2000.2 give a variance of 7.8E-3, whereas the exact true value is 0.01. Where does this inaccuracy arise?

In line 160 SIGMX ^ 2 / N is subtracted from SSQ. If you print out these values you will see that for closely spaced data they are very nearly equal. Most of the leading digits are the same, and only a few trailing digits are available to convey numerical significance to the result of the subtraction. This "subtractive cancellation" is one of the main problems in numerical computation. In general there are three (not mutually exclusive) ways to deal with the difficulty.

(1) Use more accurate floating point arithmetic. To simplify a little, the accuracy of floating point operations is governed by the number of bytes of memory allotted to each variable. In IBM machines, single precision variables have 4 bytes allotted, the same as in Apple Pascal; numbers can be represented with a precision slightly better than 1 part in a million. Applesoft serves us better! 5 bytes are allotted, giving a 256-fold improvement in precision (this is an excellent reason for preferring Applesoft to Pascal). Hayden Book Co. sells a double precision software package for the Apple, giving a fabulous 21-digit precision, but this shot-gun approach to numerical accuracy is usually not needed, as we shall see.

(2) Compute with special care the two quantities to be subtracted. In the program example, squared quantities have been calculated by the power function X^2, which is neither accurate nor fast. Applesoft interprets X^2 as EXP (2 * LOG (X)). The relatively small error in the LOG routine is magnified by the subsequent exponentiation. If we replace the power functions by direct multiplication (X * X), the program performs a little better. However, an algorithm that is upset by errors in the 8th and 9th digits cannot be called a good one. This leads us to:

(3) Use an algorithm that avoids subtraction of nearly equal quantities. Two options are available for our variance calculation. For the first, suppose the N data items to have been gathered and stored in an array X().

```
160 SIGMX = 0.0:SSQ = 0.0
170 REM COMPUTE MEAN
180 FOR I = 1 TO N
190 SIGMX = SIGMX + X(I)
200 NEXT I
210 MEAN = SIGMX / N
220 REM COMPUTE VARIANCE
230 FOR I = 1 TO N
240 TEMP = MEAN - X(I)
250 SSQ = SSQ + TEMP * TEMP
260 NEXT I
270 VAR = SSQ / (N - 1)
```

Note that the vital subtraction (now in line 240) is of values much less nearly equal than in line 160 of the first listing. In consequence this version is hardly affected by cancellation, even for data which completely flummoxes the first program (e.g. 10000, 10000.1, 10000.2; variance = 0.01 exactly). If you cannot or do not want to store the data in an array, a cumulative processing method is:

```
100 SSQ = 0.0:MEAN = 0.0:N = 0
110 PRINT "SAMPLE £" N + 1;
120 INPUT " <-9999 TO END> ? ";X
130 IF X = -9999 GOTO 180
140 N = N + 1:DX = (X - MEAN) / N
150 REM UPDATE SUM-OF-SQUARES AND MEAN
160 SSQ = SSQ + DX * DX * N * (N - 1)
170 MEAN = MEAN + DX: GOTO 110
180 REM CALCULATE VARIANCE
190 VAR = SSQ / (N - 1)
200 PRINT "MEAN = "MEAN
210 PRINT "SAMPLE VARIANCE = "VAR
220 PRINT "STANDARD DEV. = " SQR (VAR)
```

This is still not a full working routine, as we have not provided any means for data validation and correction. As the textbooks put it, "that is left as an exercise for the reader".

Two Years of Hard Core - Quick Reference

Articles and Authors 1981/1982

Reconstructed by Peter Blair and Tony Williams

February 1981 Volume 1 Issue 1

Relocating Applesoft	Michael Mathison
Personalised DOS	Ian Trackman
Input Anything	Ian Trackman
Hi Res Plotting	John Sharp
Hex Conversion	T Tse
Amateur Radio	Don Maclean
Adventurer's Friend	Keith Jones
Aims of the Group	Martin Perry
Input Routine	Geoffrey Clements
Page Zero Memory Map	David Bolton
Beginner's Page	John Sharp
Chairman's Corner	John Sharp
Business Data	Nik Spicer
Clearing Hi-Res	Michael Mathison
BASUG and IAC	John Sharp
Pippin's Page	Vernon Quaintance

April 1981 Volume 1 Issue 2

IAC Notes	
Using Apple	John Sharp
Palsoft graphics	Richard Teed
Speeding up	Graham Rubens
Chairman's Corner	David Bolton
Printer Interfacing	Chris Murphy
Beginner's Page	John Sharp
Basic Differences	John Rogers
Adventurer's Friend	Keith Jones
Pascal Page	Frank Kay
Apple 6502	Ian Trackman
Using 2716	David Bolton
Life and ITT	John Sharp
Apple II UCSD	Leo Crossfield
Pascal on ITT	Richard Teed
Eamon	Don Brown
Pippin's Page	Vernon Quaintance

June 1981 Volume 1 Issue 3

BASUG News	Tony Williams
Software Library	John Rogers
AUG Europe	Tony Williams
Disappearing It Trick	Ian Trackman
Handicapped Child	Norah Arnold
DOS Differences	John Sharp
Random Access	T Tse
Beginner's Page	John Sharp
North London Fair	Tony Williams
Visicalc	John Wingate
Visicalc	Graham Bryce
Pascal Page	Frank Kay
Printer Interfacing	Chris Murphy
Screen Formatting	Nik Spicer
Autostart	Michael Mathison
Graphics Primer	Leo Crossfield & John Sharp
War Gaming	Robin Hood
Collision Counter	Michael Mathison
Combining Applesoft and Machine Code	Neil McFerran
Pippin's Page	Vernon Quaintaince

August 1981 Volume 1 Issue 4

Editorial	David Bolton
Beginner's Page	John Sharp
Beginner Bytes Back	Tom Winkel
Introductory Disk	John Sharp
Amateur Radio	Don Maclean
DOS Append Errors	Neil McFerran
Chairman's Corner	Frank Kay
Pascal Page	Warren Avery
Park St Meetings	Tony Williams
IRAM Review	Nik Spicer
Ark and the Computer	Pete and Pam
ITT Hi-res Conversion	Ian Trackman
ITT Hi-res Mods	David Bolton
Palsoft Onerr	Ian McKeirie
Wargames	Robin Hood
Handicapped	Glyn Vernon
ACM Minutes	
Printer Interfacing	Chris Murphy
Software Libraries	John Rogers
Frances Teo: Profile	Tony Williams
Pippin's Page	Vernon Quaintance
Expediter: Review	Mike Walker

October 1981 Volume 1 Issue 5

Courses	Eddie Payne
Editorial	David Bolton
Beginner's Page	John Sharp
Education Column	Roy Garland
Sunday School	Tony Williams
Aristotle's Apple	Norah Arnold
Chairman's Corner	John Sharp
Shape Manager Review	A.P.Mullan
Shake Hands	Ian Pawson
Visicalc	Ian Trackman
How Applesoft Stores Variables	Peter Wicks
Videx View	Rev. Sqn/Ldr Cowell
DoubleDOS	John Sharp
Software Library	John Rogers
Sectal Sector	
ITT Hi-Res	Ian Trackman
Beneath Apple DOS	Chris Murphy
Apple Machine Language:Review	Ian Trackman
Books in Brief	Cliff Wooton
It's a Mystery	T Tse
More on Programs and Printers	John Rodger
Enhancer	John Sharp
Eamon Report	David Row

December 1981 Volume 1 Issue 6

February 1982 Volume 2 Issue 1

April 1982 Volume 2 Issue 2

June 1982 Volume 2 Issue 3

August 1982 Volume 2 Issue 4

more letters

Dundee

Dear Sirs,

with reference to Mr Gravil's letter in Vol 2.No2 concerning a method of printing the LR screen; there is a simpler solution available for the Paper Tiger. This machine, and some other printers, can be set in a graphics mode which will print a dot pattern corresponding to the ASCII bits of a character. Therefore it is possible to define a set of strings which will simulate the monochrome display of an LR block for each of the 16 LR "colours", use SCRN to scan the actual display and print out a facsimile of the screen.

I enclose listings for the Paper Tiger routine and one for an MPI 99G which gives a better quality print and a sample print of bars of the 15 visible "colours".

In fact it is more useful to use this technique to define your own "colours" i.e. hatchings rather than try to match the LR monochrome display and I've also enclosed some examples of hatchings which I use.

Yours

Philip Bolt

Advertising rates

Full page	£50.00
Half page	£27.50
1/4 page	£17.50
Preparation of artwork	£5.00

Please send complete camera-ready artwork in monochrome. If in A4 size then typeface must tolerate photographic reduction to A5. Or send artwork in A5. Hard Core can undertake minor alterations to copy.

Editorial copy to 2, Nicholas Gardens, London W5 5HY

DIARY

November 25/27th Northern Computer Fair, Bellevue Manchester

December
1st Leicester Laughs AGM + Synthesizer
4/5th BASUG Beginning Machine Code Course with Ian Trackman
7th Herts Group Meeting
9th South London Group "Games"
11th BASUG Workshop - Word Processing + Bring & Buy Sale.
Kenilworth Hotel, Gt Russell St. £1 only. Bring no machines.
14th Kent's Own meeting
17th Apple Medical Forum + BASUG Medical SIG (Middlesex Hospital)

January 1983
29th/30th Weekend Event, Milton Keynes 1 day on File Handling 1 day Beginners Basic + workshop.

February
3rd Leicester LAUGHS "Hard Disks"+"Languages for the Apple"

March 3rd Leicester LAUGHS "Printers"

(What dates does your group want to publicise?)